# [***1.1. Perform an Initial Data Analysis***](https://openclassrooms.com/en/courses/6037301-perform-an-initial-data-analysis)

# **Identify Different Types of Errors**

Welcome!

In this course, you will be learning how to **cleanse** and **describe** your data.

When you begin analyzing your data, you need to cleanse them of all errors. If not, the code you write to make your lovely graphs (and other representations) will crash. Worse, if your sample contains errors, your analyses may well contain errors, too.

Every data analyst or data scientist will tell you that, unfortunately, we spend more time cleansing our data than analyzing them. Cleansing your dataset is a crucial component in data analysis, even though it may not be the most exciting part. :honte:

It would be false to say that data cleansing is performed before data analysis. In most cases, we have to go back and forth between the cleansing phase and the description (analysis) phase. During the analysis phase, we often encounter new errors, and are forced to go back again and cleanse. In addition, the cleansing required for analysis differs from one statistical process to the next: that’s why you have to go back and forth!

Wait, why do we have errors in the first place?

It’s all comes down to a question of *data source*.

If you have data input by humans, it is highly likely to have errors. Imagine you are volunteering as a poll worker for a local election and are asked to gather elections results on paper and then manually type them into a spreadsheet program. At some point, some way, an error is bound to occur!

Consider another situation where data is provided by sensors (for example, a geolocation system on your cell phone, a thermometer, or speed sensors on your car). An error can occur if the sensor has deteriorated over time and is no longer or no longer works at all (is no longer sending data).

### ***1.2.Identify the Different Error Types***

We are going to look at a few different types of errors. There is no need to learn them by heart or remember their names: these details aren’t important!

Let’s take the example of a sample of people described by a number of different variables:

| **First Name** | **Email** | **Date of Birth** | **Country** | **Height** |
| --- | --- | --- | --- | --- |
| Leila | leila@example.com | 23/01/1990 | France | 1.49m |
| Samuel | samuel\_329@example.com | 20/09/2001 |  | 1.67m |
| Rodney | choupipoune@supermail.eu | 12 Sept. 1984 | Madagascar | 5'2 |
| Mark | marco23@example.com, mc23@supermail.eu | 10/02/1978 | 24 | 1.65m |
| Harry | helloworld@mail.example.com | 04/25/1975 | Germany | 1.34m |
| Hannah | hannah2019@supermail.eu | 01/01/1970 | Canada | 2.8m |
| Samuël | samuel\_329@example.com |  | Benin | 1.45m |

Well... You can see that this sample is not super clean, right?

Can you point out a few inconsistencies? Write them down a few and check your answers below!

1. First, there are empty cells for the "country" and "date of birth variables". We call these **missing attributes**.
2. If you look at the "Country" column, you see a cell that contains 24. “24” is definitely not a country! This is known as a **lexical error**.
3. Next, you may notice in the "Height" column that there is an entry with a different unit of measure. Indeed, Rodney's height is recorded in feet and inches while the rest are recorded in meters. This is an **irregularity error** because the unit of measures are not uniform.
4. Mark has two email addresses. It’s is not necessarily a problem, but if you forget about this and code an analysis program based on the assumption that each person has only one email address, your program will probably crash! This is called a **formatting error**.
5. Look at the "date of birth" variable. There is also a **formatting error** here as Rodney’s date of birth is not recorded in the same format as the others.
6. Samuel appears on two different rows. But, how can we be sure this is the same Samuel? By his email address, of course! This is called a **duplication error**. But look closer, Samuel’s two rows each give a different value for the "height variable": 1.67m and 1.45m. This is called a **contradiction error**.
7. Hannah is apparently 9'1". This height diverges greatly from the normal heights of human beings. This value is, therefore, referred to as an **outlier**.

The term **outlier** can indicate two different things: an **atypical** value and an **aberration**.

### **Deal With These Errors**

I’ll tell you right away that, when it comes to cleansing data sets, there is no set rule. Everything you do depends on how you plan to use your data. No two data analysts will cleanse the same data set the same way—not if their objectives are different!

So there’s no set rule, but I can give you a few pointers:

1. 1. Missing attributes will be addressed in the following chapter.
2. 2. For the invalid country, it’s possible to supply a list of authorized countries in advance, then eliminate all of the values that are not found on this list (hint: 24 will not be found). Such a list is often referred to as a dictionary.
3. For irregularity errors, it’s more complicated! You can, for example, set a fixed format (here: a decimal number followed by the letter “m” for “meter”) and eliminate values that don’t adhere to it. But we can do better, by first detecting what unit the value is expressed in (meters or centimeters) then converting everything to the same unit.
4. For the formatting error of the duplicate email address, it all depends on what you want to do. If you won’t be looking at emails in your future analysis, there’s no need to correct this error. If, on the other hand, you want to know the proportion of people whose address ends in, for example @example.com, or @supermail.eu, etc., then you can choose between:
   1. Taking the first email address and forgetting the second one.
   2. Keeping all email addresses.
5. Let’s move on to the Date of Birth variable. Aaaaaaah, dates! Believe me, they will always give you headaches! There are many different formats; each country has its own custom when it comes to writing dates (France and North America, for example, do not use the same format). Add to this the problem of time zones! In our case, the simplest solution would be to eliminate dates that are not in the desired format month/day/year.
6. Duplicates will be discussed in the next chapter.
7. Outliers will also be discussed in the next chapter!

Some countries have come together and agreed upon a uniform standard for dates. It’s called the ISO 8601 format, and it looks like this: 1977-04-22T06:00:00Z.

As a general rule, if a variable contains few errors and the variable is not of crucial importance to your analysis, you can allow yourself to delete erroneous values. You will then be left with missing attributes. You will see what to do with these in the next chapter.

However, if there are many errors of the same type, you might as well create a program to correct them. For example, if 60% of the heights are expressed in meters, 35% in centimeters, and 5% in other units, then 35% of the errors are of the same type (35% of the values are expressed in centimeters instead of meters). In this case, you might as well write a few lines of code to convert the centimeters to meters. If you are motivated and the data set is worth it to you, go ahead and correct the remaining 5%, although that might take you longer!

### **Take It Further: External Resources**

In this course, we will cleanse our data using Python.

But you should know that there is a very good tool for cleansing data, accessible to non-programmers, called [OpenRefine.](http://openrefine.org/)

## **1.3. Deal with Missing Attributes, Outliers, and Duplicates**

In the last chapter, we saw that a sample can contain missing values, outliers, and duplicates. What to do?

Some of the following methods delete information from your sample, so be sure to always make a copy of your sample. If you are thinking of deleting information, make a copy of the sample, then delete whatever you want: the new sample you obtain will be called a sub-sample (or subset).

### **Missing Attributes**

When the sample contains missing attributes, there is unfortunately no miracle cure! However, there are several possible approaches you can take.

For a given variable (for instance, last chapter’s example of Date of Birth), if the proportion of missing attributes is low, you can just forget about them and do nothing: leave the sample intact. You will then be working with a data set that has “holes,” like a Swiss cheese. Depending on the statistical process you plan to apply, this solution might be acceptable, or it might not.

#### **Forget a Variable**

However, if for this same variable, the proportion of missing attributes is way too high, you’d better just forget about it—provided that the variable is not too important to your analysis. This is the same as not including a column in a table, as we saw in the last chapter.

#### **Forget Individuals**

If the variable with the missing data is crucial to your analysis, it’s better to create a sub-sample, removing the individuals for whom this variable is missing. For example, if you are analyzing your bank statements by looking at the amount of money you spend/earn, the “transaction amount” variable will be crucial. If the transaction amount is unknown for some of the rows of your statement, it’s better to create a sub-sample that removes all of the offending rows.

However, this method is risky. You might find yourself with a number of individuals (a number of rows) so small that your analysis no longer meaningful. In addition, your sample might no longer be representative of the overall population. To find out why, go to the Take It Further section at the end of this chapter.

#### **Guess**

A more adventurous approach consists of filling in your holes with values you have guessed. This is pretty much the method for daredevils! :zorro: Of course, these values will not correspond to actual values, but some methods manage to create values that are not too far off. Guessing a missing attribute is referred to as Imputation.

For example, we can replace the missing attributes of the height variable with the average height of the individuals in our sample. In our example, to correct Hannah’s heiDeal with Missing Attributes, Outliers, and Duplicates

In the last chapter, we saw that a sample can contain missing values, outliers, and duplicates. What to do?

Some of the following methods delete information from your sample, so be sure to always make a copy of your sample. If you are thinking of deleting information, make a copy of the sample, then delete whatever you want: the new sample you obtain will be called a sub-sample (or subset).

**Missing Attributes**

When the sample contains missing attributes, there is unfortunately no miracle cure! However, there are several possible approaches you can take.

For a given variable (for instance, last chapter’s example of Date of Birth), if the proportion of missing attributes is low, you can just forget about them and do nothing: leave the sample intact. You will then be working with a data set that has “holes,” like a Swiss cheese. Depending on the statistical process you plan to apply, this solution might be acceptable, or it might not.

**Forget a Variable**

However, if for this same variable, the proportion of missing attributes is way too high, you’d better just forget about it—provided that the variable is not too important to your analysis. This is the same as not including a column in a table, as we saw in the last chapter.

**Forget Individuals**

If the variable with the missing data is crucial to your analysis, it’s better to create a sub-sample, removing the individuals for whom this variable is missing. For example, if you are analyzing your bank statements by looking at the amount of money you spend/earn, the “transaction amount” variable will be crucial. If the transaction amount is unknown for some of the rows of your statement, it’s better to create a sub-sample that removes all of the offending rows.

However, this method is risky. You might find yourself with a number of individuals (a number of rows) so small that your analysis no longer meaningful. In addition, your sample might no longer be representative of the overall population. To find out why, go to the Take It Further section at the end of this chapter.

**Guess**

A more adventurous approach consists of filling in your holes with values you have guessed. This is pretty much the method for daredevils! :zorro: Of course, these values will not correspond to actual values, but some methods manage to create values that are not too far off. Guessing a missing attribute is referred to as Imputation.

For example, we can replace the missing attributes of the height variable with the average height of the individuals in our sample. In our example, to correct Hannah’s height (which we assume is erroneous), we would replace it with the average height of the other individuals in the sample, which is 1.52 m. This is known as Mean Imputation.

**Guess Based on Other Variables**

But we can do better! To replace a given variable, we can look at the variables around it. A number of methods apply this principle.

Imagine a new individual named Luke, born in 1991, whose height is unknown. Rather than assign him the mean of the entire sample (1.52 m), we can assign him the average height of people who are about his age. So let’s assign him the average height of people born between 1990 and 2000, or 1.49 m. Here, we looked at the value of the date\_of\_birth variable to come up with a value for the height variable.

Other methods also deduce the value of a variable by looking at other variables. These include Hot-deck, and methods based on linear regression.

Guessing (imputing) values changes your sample, because the imputed values are false. In particular, your calculations of variances and correlations will be false. You must therefore use this method only sparingly.

In all cases, you must specify which method you used for each of the analysis results you present. It’s a question of intellectual integrity ^^.

**Outliers**

Hannah is 3.45 meters tall. You think that’s not so tall? You’re wrong. It’s very tall compared with the heights of other human beings.

But proceed cautiously, because an outlier value isn’t always necessarily false! Hannah might actually be 3.45 meters tall. Okay, that’s hard to believe—but it’s possible.

An outlier can be:

An aberration: a value that’s obviously false

An atypical value: a value that “deviates from the norm,” but is not necessarily false.

Ideally, outliers should be checked to determine whether or not they’re erroneous. For example, a thermometer in Canada in April might indicate 40°C, but this could be due to a defective temperature sensor, or it could be an actual value....(although it usually a little colder in Canada in the spring ;)).

So what should we do with outliers? If we are sure that the value is erroneous (input error or flawed sensor, for example) and we can’t find the actual value, it has to be deleted. If we are not sure whether it’s erroneous, we can choose between:

Deleting the value. We then find ourselves with a missing attribute, to which we can impute a value, as we saw previously. But imputation isn’t mandatory.

Keeping the value.

How to choose between these two options? It all depends on how you will be processing your data after that. Some methods are considered “robust,” meaning they are not destabilized by outliers. For example, we will see below that the mean is very sensitive to outliers, while the median is not. So if you want to find a mean, create a sub-sample in which you don’t include outliers. But if you also want to calculate a median, work with the original sample.

When you present your analysis, don’t hesitate to point out outliers if they are interesting. For example, “The temperature statements contain two outliers of 42°C, corresponding to two days of extreme temperatures.”

What about Duplicates?

In our example, Samuel appears twice. That’s a problem, because this duplicate compromises the analysis, in particular by falsifying the sample’s average height.

Duplicates have to be removed. However, there is no precise rule for detecting them: you alone can find them, based on the structure of your data and your knowledge of how the data were collected. But sometimes, it will be impossible.

A little example: if your sample contains an “identifier” variable, then it’s easy to detect duplicates. They are the ones with the same identifier . In our example, we can consider the email address to be an individual’s identifier. In our example, the two rows containing the email address samuel\_329@example.com constitute a duplicate.

If you are familiar with databases, you are probably familiar with the concept of keys (primary keys, candidate keys). Two individuals that have the same value for a key constitute a duplicate!

Another example: Say you are analyzing temperature records from a small town. The town has two weather stations. Station 1 operated for many years until January 15, 2019, and then was taken offline, due to age. Because this was expected, Station 2 had already been installed (in the same place) to take over for it. Station 2 began operating on January 2, 2019. Your sample is therefore made up of records from both stations. However, records made between January 2 and January 15 are duplicates, because both stations were operating at the same time. For each date in this interim period, therefore, you must delete one of the two records.

Yes but which of our two rows containing samuel\_329@example.com do we delete? Do we just pick one at random?

Cases like this call for greater attention. It’s better, in fact, to group the duplicates in one row. Of the two rows in our sample, the first informs us that Samuel was born on 20/09/2001, and the second informs us that Samuel lives in Benin (information that is missing from the first row). The real problem has to do with the height: the first row tells us that Samuel is 1.67 meters tall, while the second tells us that he’s only 1.45 meters tall. That’s a contradiction. If there is no other means of verifying Samuel’s height, we can, for example, choose to take the mean of these two values.

Take It Further: The Consequences of Removing Individuals

Imagine a sample of people represented in the same form as in the last chapter:

First Name Country Date of Birth Height

Albert France 23/09/1930 1.45m

Sophia USA 01/20/1959 1.68m

Donald USA 02/16/2002 1.65m

Ali France 16/02/2000 1.57m

Doriane Togo 17/08/1978 1.58m

You decide to delete all of the dates of birth that don’t conform to the format day/month/year, which creates missing attributes for the date of birth variable. Then you decide to delete all of the rows (all of the individuals) that have a missing date of birth. You will probably end up removing all of the people who live in the United States, because their dates are expressed in a different format from those of French-speaking countries. If you then perform an analysis on the heights, your sample will no longer be representative, because people from the United States surely have a different average height than those of other countries.

# **1.4. Cleanse Your Dataset using Python**

We are now going to cleanse the data set we saw in the previous chapter. We will illustrate this in Python.

We will begin by loading the sample [persons.csv](https://s3-eu-west-1.amazonaws.com/static.oc-static.com/prod/courses/files/Data+Analyst+EN%C2%A0Path/Courses/Perform+an+Initial+Data+Analysis/persons.csv) (which you can find here) into a variable we will call data . This variable will therefore be a dataframe.

Next, we will comb through each of the columns looking for errors, correcting them, and updating the columns accordingly. Whether you are working in Python or R, updating a column in a dataframe is performed like this:

data["name\_column"] = new\_column

Here, we want to replace the values of the name\_column column (or variable). If the

dataframe has 7 lines, then name\_column column will contain 7 values. To replace them,

new\_column has to be a list of 7 values.

### **Use the Apply and Map Methods**

We still need to know how to populate new\_column . In fact, this will be calculated from

name\_column . We need to comb through each name\_column value, verify whether it is correct or not, and correct it as needed. For this, we use the apply method. This method applies a function to each value of a dataframe column. Alternatively, we can use the map method, which is ([more or less](https://stackoverflow.com/questions/19798153/difference-between-map-applymap-and-apply-methods-in-pandas)) equivalent. It applies a check / correct function to each value:

import pandas as pd # the Pandas Libraries is imported and aliased 'pd'.

def lower\_case(value):

print('Here is the value I am processing : ', value)

return value.lower()

data = pd.DataFrame([['A',1],

['B',2],

['C',3]], columns = ['letter','position'])

new\_column = data['letter'].apply(lower\_case)

new\_column = new\_column.values

print(new\_column)

data['letter'] = new\_column

print(data)

This code is provided in the cleanse folder of the downloadable archive in the Download the data chapter, in the following separate files: R\_cleanse and python\_cleanse.

On line 7, we create our dataframe. This is a table with 2 columns ('letter' and 'position') and 3 rows. On line 3, we create a function named lower\_case , which takes as a parameter a value , displays it (line 4), converts it to lower case (line 5), then returns it. Next, we select the letter column from data , call the apply method, and specify that each of the values be sent, one at a time, to the lower\_case function (line 13).

On line 11, new\_column is “column” type (because the apply method returns a column). In the Pandas library, the exact column type is Series . To obtain the values for this column in the form of a list, we call new\_column.values (line 12). Here is what the program will display:

Here is the value I am processing : A

Here is the value I am processing : B

Here is the value I am processing : C

['a' 'b' 'c']

letter position

0 a 1

1 b 2

2 c 3

Lines 1 to 3 display what the lower\_case function is doing; line 4 displays the processing result, that is, the three lower-case letters; and the other lines display the dataframe in which the letter column has been updated to lowercase!

Line 12 converts new\_column from a Series object to a list. This line is optional, in fact, because the syntax of line 13 works just as well whether new\_column is a list or a Series object!

### **Attack!**

#### **Load the Data**

Begin by downloading the CSV file that corresponds to the example in previous chapters (provided at the beginning of the chapter), then load it using these lines of code:

# importation of libraries we will need

import pandas as pd

import numpy as np

import re

# loading and display of data

data = pd.read\_csv('persons.csv')

print(data)

#### **Process Country Names**

As you have no doubt understood, we will need one function per process. Let’s forget lower\_case , and write a function that verifies whether the countries in the Country column are correct. To do this, we need a list of valid country names:

VALID\_COUNTRIES = ['France', 'Madagascar', 'Benin', 'Germany'

, 'Canada']

def check\_country(country):

if country not in VALID\_COUNTRIES:

print(' - "{}" is not a valid country, we delete it.' \

.format(country))

return np.NaN

return country

Here, if the country in the country variable is not on the VALID\_COUNTRIES list, we display that message on lines 6 and 7. Then, we return np.NaN , which is the value used by the Numpy and Pandas libraries to indicate that a value is unknown. It is roughly equivalent to None.

More specifically, "NaN" means "Not a Number.” You will also see pd.NaT , "Not a Time" (for date columns).

Otherwise, if the country is valid, we simply return it (line 9)!

#### **Process Emails**

Now it’s the emails’ turn! The problem with this column is that there are sometimes two email addresses per row. We only want to take the first one. We will therefore create the first function:

def first(string):

parts = string.split(',')

first\_part = parts[0]

if len(parts) >= 2:

print(' - There are several parts in "{}", we are only keeping {}.'\

.format(parts,first\_part))

return first\_part

When there is more than one email per line, they are separated by commas. We therefore separate the character string of the string variable according to commas using the split method (line 2). The result is a list with as many items as there are email addresses; this list is placed in the parts variable.

Since parts contains at least one item, we place it in the first\_part variable. Then we count the number of items in parts using the len function. If there are at least two items, we display the message shown on lines 5 and 6. Finally, we return first\_part , which contains the first email!

#### **Process Heights**

Here we will have two functions: convert\_height , which will convert character strings of type 5'4 to decimal numbers, and fill\_height , which will replace the missing attributes with the average height (mean) of the sample.

def convert\_height(height):

found = re.search('\d\.\d{2}m', height)

if found is None:

print('{} is not in the right format. It will be ignored.'.format(height))

return np.NaN

else:

value = height[:-1] # the last character is removed: 'm'

return float(value)

def fill\_height(height, replacement):

if pd.isnull(height):

print('Imputation by the mean : {}'.format(replacement))

return replacement

return height

The first function is a little more elaborate. You can just blindly trust it, or you can attempt to pierce its veil of mystery by reading the *Take It Further* section at the end of this chapter. :magicien:

Let’s move on to the second function. Ah! It takes two parameters: height and replacement . The first is the height, as usual. The second is the value to be returned if there is a missing attribute. Line 11 checks to see whether the height value is missing (None, NaN or NaT). If it is, we return the replacement value (line 13). Otherwise, we return height .

#### **Apply All Functions**

Now that these functions are defined, let’s execute them! At the end of your program, add the following:

data['email'] = data['email'].apply(first)

data['country'] = data['country'].apply(check\_country)

data['height'] = [convert\_height(t) for t in data['height']]

data['height'] = [t if t<3 else np.NaN for t in data['height']]

mean\_height = data['height'].mean()

data['height'] = [fill\_height(t, mean\_height) for t in data['height']]

data['date\_of\_birth'] = pd.to\_datetime(data['date\_of\_birth'],

format='%d/%m/%Y', errors='coerce')

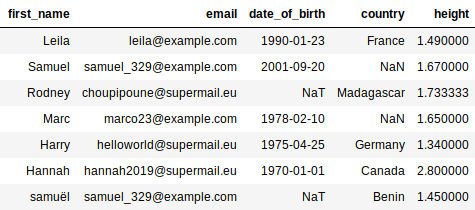
print(data)

Do you recall the syntax for updating a column we saw at the very beginning of this chapter? We use it here in lines 1 to 4, 6 and 7. You are familiar with the syntax used in lines 1 and 2. However, for lines 3, 4 and 6 you may need to refresh your memory about list comprehensions. If this term means nothing to you, scroll down to the *Take It Further* section.

What else is there left to tell you? You have all of the essentials. Well, except for a few minor details:

* h if h<3 else np.NaN returns h if h is less than 3, otherwise it returns np.NaN . This is used to delete heights in excess of 7'0", which are aberrations.
* data['height'].mean() returns a unique value, which is the mean of all the heights.
* The date\_of\_birth column contains character strings. We convert them to dates, specifying the date format. Character strings that do not conform to this format will be converted in pd.NaT (this is the case for Rodney’s date of birth).

Line 9 displays the final result:



### **Take It Further: List Comprehension**

List comprehension is a very practical syntax, because it can be used to write a loop that creates a list, in one line. For example, line 3 of the last bit of code above is equivalent to 4 lines (lines 3 to 6):

data = pd.read\_csv('persons.csv')

new\_column = []

for t in data['height']:

new\_column.append(convert\_height(t))

data['height'] = new\_column

Would could very well have used apply :

data = pd.read\_csv('persons.csv')

data['height'] = data['height'].apply(convert\_height)

### **Take It Further: Processing heights**

Let’s go back to the function we left behind:

def convert\_height(height):

found = re.search('\d\.\d{2}m', height)

if found is None:

print('{} is not in the right format. It will be ignored.'.format(height))

return np.NaN

else:

value = height[:-1] # the last character is removed: 'm'

return float(value)

Normally, Pandas automatically detects whether a column from a CSV file contains numbers or character strings. But here, the Height column contains "m"s (for “meters”). Because “m” is a letter, Pandas considers "1.34 m" to be a character string, not a number! We must therefore convert it ourselves.

So... it’s true that line 2 is hard to understand. It checks to see whether the height is properly formatted, that is, in the form of a number followed by a decimal point, then two numbers, then an "m." Thus, "1.34 m" is correct; "153 cm" is not.

This is a regular expression. No need to learn about them for this course, but you should know all the same that regular expressions are very practical—indispensable if you want to become a Data Analyst.

You now hold the keys to understanding the rest of this function. Note that float(value) is used to convert a character string representing a number to... a real number (whose type is “float”)!

**1.5. Quiz**

**Question 1**

The following table shows the Rainfall (in mm) and the Average Temperature (in °C) for each city over the month of **April**.

| **City** | **Country** | **Rainfall (mm)** | **Average Temperature (°C)** |
| --- | --- | --- | --- |
| Vancouver | Canada | 100.8 | 9.8 |
| Bogota | Colombia | 90.1 | 15.2 |
| Queenstown | New Zealand | 67 | 11 |
| Paris | France | 25 | 32 |
| Mombasa | Kenya | 150 | 27 |

Taking a look at this, you notice that the *average temperature* for *Paris, France*, seems abnormally high! In previous years, the average temperature in April has been much lower. You are unsure whether or not the value is erroneous.

What are your options?

Delete the value and impute another value

Decide that the statistical treatment you are using is robust (i.e. not sensitive to outliers), and keep the value.

Decide that the statistical treatment is not robust, and delete the value.

All of the above

**Question 2**

**Examine this block of code. What does it do?**

import pandas as pd

STATUS\_VALUES = ["GUEST","EMPLOYER","EMPLOYEE"]

df = pd.read\_csv("mylittlecompany.csv")

def process(value):

if value not in STATUS\_VALUES:

return "GUEST"

else:

return value

df["status"] = df["status"].map(process)

They assign the value “GUEST” to individuals that do not have the value “EMPLOYER” or “EMPLOYEE” for the “status” variable.

**Question 3**

**Which column in this table contains an error of irregularity?**

| identifier | age | first name | score |
| --- | --- | --- | --- |
| 2873 | 27 | Leila | 39 points |
| 1028 | 999 |  | 45 points |
| 3892 | 78 | Samir | 89% |
| 8273 | 12 | Cindy | 24 points |

Score

**Question 4**

What is TRUE about duplicates?

Careful, there are several correct answers.

Duplicates often compromise our analysis.

If a sample contains an "identifier" variable, then it is easier to detect duplicates.

**Question 5**

What is TRUE about imputation?

Careful, there are several correct answers.

When you impute a value, you assign a value to a missing value in a sample.

When you impute a value, you must always specify which method you used to do so.

**Question 6**

Which column in this table contains an outlier?

| **Identifier** | **Age** | **First Name** | **Score** |
| --- | --- | --- | --- |
| 2873 | 27 | Leila | 39 points |
| 1028 | 999 |  | 45 points |
| 3892 | 78 | Samir | 89% |
| 8273 | 12 | Cindy | 24 points |

Age

**Question 7**

What statement is FALSE about this table?

| **Identifier** | **Age** | **First Name** | **Score** |
| --- | --- | --- | --- |
| 2873 | 27 | Leila | 39 points |
| 1028 | 999 |  | 45 points |
| 3892 | 78 | Samir | 89% |
| 8273 | 12 | Cindy | 24 points |

The column "First Name" has a missing value.

We can perform a mean imputation in the column "First Name".

There is an irregularity error in the "Score" column.

**Question 8**

What would be the result of an imputation by the mean on this data :

| **Identifier** | **Age** | **Score 1** | **Score 2** |
| --- | --- | --- | --- |
| 45 | 20 | 12 | 39 |
| 46 |  | 44 | 42 |
| 47 | 60 | 13 | 12 |
| 48 | 39 | 56 | 21 |
| 49 | 41 | 34 | 34 |

**Step 1: Identify missing values:**

* **Age** is missing for row 2.

### **Step 2: Calculate the mean for the Age column (excluding the missing value):**

Available values in **Age** = 20, 60, 39, 41

20+60+39+414=1604=40\frac{20 + 60 + 39 + 41}{4} = \frac{160}{4} = 40420+60+39+41​=4160​=40

✅ **Missing Age** in row 2 = **40**

40

# **2.1.Adopt the Basic Terminology Used in Statistics**

Now that we have cleaned our data, we need to know how to represent it!

Before we dive in head first, I'm first going to provide an overview of statistics.

### **An Overview of Vocabulary**

In statistics, we study thingamajigs, whatchamacallits, and other things.

Awesome. Thanks for telling me! Anything else?

Let me break it down for you! These “things” are known in statistics as **individuals**. Individuals can be objects, people, animals, physical measurements, etc. An individual is a *unit of observation*.

Individuals have characteristics: these are called characters, attributes, or **variables**.

The set of all of the individuals is called a **population**. Its size is often denoted by N , which corresponds to the number of individuals in the population. The exact size of a population is often unknown.

A subset of individuals within a population is called a **sample**. Its size is often denoted by n.

The term **dataset** will be used over and over again. This term has no exact definition, but for the purposes of this course it will be equivalent to a sample.

How is a sample represented?

In general, a sample is represented in the form of a table, in which each row corresponds to an individual, and each column represents a variable. This representation forms the basis of the file format known as [CSV](https://en.wikipedia.org/wiki/Comma-separated_values) (Comma-Separated Values). CSV files can be opened in a spreadsheet program (Microsoft® Excel, OpenOffice Calc), and are easily interpreted by R and Python.



Representation of a sample

This representation is very similar to the one used for [**relational databases**](https://openclassrooms.com/en/courses/2071486-retrieve-data-using-sql).

### **An Overview of Statistics**

#### **Understand the Difference Between Statistics and Probability**

Statistics and probability are the same thing, aren’t they?

Uh... no! The two fields are certainly closely related, but they are distinct from one another. When all you are doing is observing and describing objectively, that’s **statistics**.

But as soon as you start modeling, making connections between observation and theory, that’s **probability**: at that point, you’ve moved into the world of **inferential statistics**.

In statistics, the data you observe are called **observations**, or sometimes **realizations**. Based on these observations, you can form models. Statistical modeling is a way of determining the mathematical rules governing the data you observe. With probability, you work with random variables, laws of probability, etc.

For example, to study the proportion of men to women in a country, you select a sample in which you observe a proportion of 55% women to 45% men. These are *statistics*.

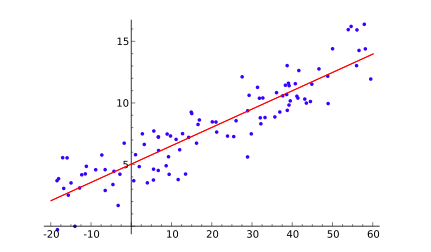
But if you then say, in this country, each child that is born has a 55% probability of being female, now you are working with *probability*!

In this course, there will be no “probs,” only stats!

#### **The Different Branches of Statistics**

##### **Descriptive Statistics**

Descriptive statistics is what this course is all about! It deals with the presentation, description, and summarization of data sets using graphs and measures (such as the mean, standard deviation, etc.). In descriptive statistics, each graph (or measure) is calculated on the basis of one or two variables at a time, no more. Why? Because it is fairly simple to represent the relationships between two variables on a graph, both on paper and on a screen, because you’re only working in two dimensions (length and width).



Two-dimensional Graph with Horizontal and Vertical Axis (source: Wikipedia)

##### **Exploratory Data Analysis**

Exploratory data analysis is an extension of descriptive statistics, except that here, you study the relationships between three or more variables. Graphs with 3, 4, 5, or 100 dimensions can’t be represented on two-dimensional paper. We need special techniques to further describe and explore the data.

##### **Inferential Statistics**

Inferential statistics deals with analyzing data relating to a sub-population in order to infer the characteristics of the population as a whole. If one day you find yourself talking about **estimators** and **hypothesis testing**, you’ll be working with inferential statistics.

##### **Statistical Modeling**

Statistical modeling involves observing the characteristics of a sample, then formalizing these observations according to mathematical rules. This formalization is called a **probabilistic model**. When you describe a phenomenon using a model, you can make forecasts or predictions.

### **Take It Further: Data Analyst vs. Data Scientist**

But what’s the difference between a Data Analyst and a Data Scientist?

The distinction between these two professions is somewhat hazy, but Data Analysts can be said to work with descriptive, exploratory, and inferential statistics. Data Scientists must master all three of these disciplines, but must also be capable of ***modeling*** phenomena. They have a battery of algorithms at their disposal that enable them to find the most effective model for the problem they are analyzing.

**2.2.Discover the Four Variable Types**

Variables come in two types, each of which is subdivided into two groups. Hence, four variable types ;).

### **Quantitative Variables**

These are variables that take a numerical value (in other words, a number!)—provided this numerical value represents a mesurable quantity and mathematical operations can be applied to it.

For example, if you add together all of the expense amounts in your bank statement, you will get the total amount you have spent.



This makes sense. It’s an amount of money. If you were to add up the amounts of each transaction, you would know that you spent $267.15 between 06/10/2023 and 06/14/2023.

However, a transaction identifier, despite being numerical, is not a **quantitative** variable. It makes no sense to add up all of your transaction identifiers as the identifier is not a measureable quantity. I shouldn’t be telling you this (it will spoil the surprise!), but an identifier is a **qualitative** variable (shhhh, you didn’t hear it from me!).

A quantitative variable is either **discrete** or **continuous**.

Discrete variables have a **countable** number of values between any two values. Think of a digital clock. A discrete value for time could be 5:34pm. This is discrete because the next value is 5:35pm.

Continuous variables have an **infinite** number of values between any two values. This time, think of an analog clock. We could have hours and minutes, but we can also include seconds, milliseconds, microseconds and so forth. In other words, there are no breaks between two values.

In the wonderful world of computers, a variable is never really continuous. To represent a continuous variable, your computer would have to store a number with an infinite number of digits following the decimal, which isn’t possible.

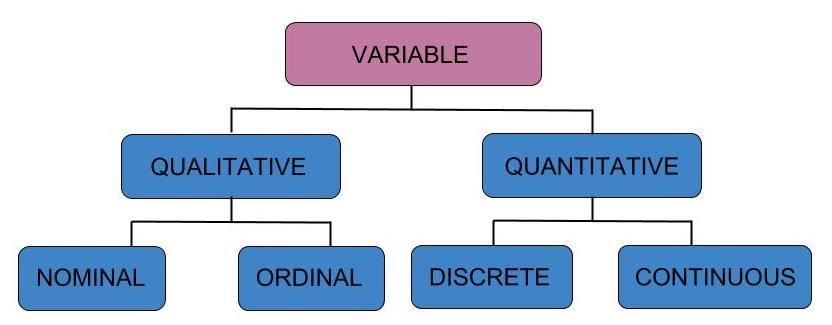
Our “debit” variable is *technically* discrete (there’s no possible value between $1.22 and $1.23), but we consider it continuous anyway, because a difference of one penny is fairly negligible.

### **Qualitative Variables**

A qualitative variable is any variable that is not quantitative . The values a qualitative variable can take are called **classes**, or **categories**. Categories are expressed in literal form (by word, phrase, or code) or by numerical codes to which mathematical operations *cannot* be applied.

A qualitative variable is either **nominal** (or **categorica**l) or **ordinal**.

A variable is ordinal if its categories can be ordered. For example, traditional exam grades (A, B, C) are **ordinal** because A is considered to be a better score than B, which is considered to be a better score than C. The transaction identifier is a **nominal** (or **categorical**) variable, because transaction no. 1 is not necessary “less” or "more" than transaction no. 40 (this is assuming the identifiers are not necessarily ordered by transaction date)



Types of variables

And there you have it: now you know everything! :zorro:

Oh, wait... the so-called **dichotomous** variables. We have yet to talk about those. Well, these are *qualitative* variables that take on only *one of two* categories such as yes/no or true/false . These are also called **binary variables**, or Booleans (abbreviated “bool”).

### **Take It Further: Are Dates Quantitative or Qualitative?**

... What about dates?

Computer systems store dates in the form of integers called *timestamps*. Timestamps count the number of seconds (and sometimes milliseconds) that have ticked away since January 1, 1970. For example, the date of September 23, 2020, is coded as follows by the timestamp: 1600819200 .

However, since adding timestamps together makes no sense, a date is considered a qualitative variable.

...Is it ordinal or nominal?

Well, timestamps are ordered. For example, 1600819200 is one second before 1600819201. We would therefore consider timestamps to be *ordinal qualitative variables*.

If you see the date 01/01/1970 in your dataset, then it most likely indicates that information is missing, and that the timestamp 0 was assigned by default!

# **2.3.Represent the Empirical Distribution of a Variable**

First thing's first: download a file called [operations.csv](https://s3-eu-west-1.amazonaws.com/static.oc-static.com/prod/courses/files/Data+Analyst+EN%C2%A0Path/Courses/Perform+an+Initial+Data+Analysis/operations.csv). This is a fictitious statement in which parts of the transaction labels have been redacted.

Second, load the dataset:

data = pd.read\_csv("operations.csv",parse\_dates=[0])

**And we're off! Begin by opening the file.**

### **Understand the Data and the Code**

Every bank statement contains at least these three pieces of information (for each transaction):

* date of transaction
* description of transaction
* amount of transaction

We are going to use these three pieces of information to create some variables:

**transaction\_date  
label  
amount  
debcr**: indicates whether the transaction is a debit or credit  
**balance\_bef\_transaction**: account balance before the transaction was made  
**categ**: transaction category, for example: “groceries,” “rent,” “bill,” etc.  
**type**: transaction type, for example: “deposit,” “charge,” “withdrawal," etc.  
**expense\_slice**: indicates whether an expenditure is small, medium, etc.  
**year**: year as determined by the transaction\_date  
**month**: month as determined by the transaction\_date  
**day**: date of the month (between 1 and 31)  
**day\_week**: day of the week (Monday, Tuesday, etc.)  
**day\_week\_num**: number of the day of the week (between 1 and 7)  
**weekend**: indicates that the transaction occurred on a weekend  
**quart\_month**: value 1, 2, 3 or 4, indicating how far into the month the transaction occurred (1: beginning ... 4: end)  
**wait**: for each transaction in the “groceries” category, indicates how much time has elapsed (in number of days) since the previous “groceries” transaction. This will be calculated later, in the chapter on linear regression.

How can a variable be represented?

So far, we have seen how to display a sample (in the form of a table in which each row represents an individual, and each column represents a variable). To represent the *categ* variable, for example, we could select the table’s *categ* column, and display it as follows:



But, you have to admit, it’s not very readable! Moreover, a sample can contain 1,000 individuals or more. A column with 1,000 values is not pretty, and it’s really hard to interpret. There’s a much better way, which consists of saying:

*A value for GROCERIES occurs 39 times, a value for OTHER occurs 212 times, a value for TRANSPORT occurs 21 times, etc.*

This formulation is called an **empirical distribution**, which we represent graphically.

### **Represent an Empirical Distribution**

The various “possibilities” that can be observed for the *categ* variable are its **categories**. Here, the categories of the *categ* variable are: *groceries, transportation, other, rent, etc.* For quantitative variables, however, these are referred to as possible **values**. Each category (or value) is associated with a number of **occurrences**. The number of occurrences for the *groceries* category is n groceries = 39.

If we divide an occurrence by the number of individuals in the sample (represented by n ), we obtain a **frequency**.

The empirical distribution of a variable is all of the values (or categories) taken by this variable, and their associated occurrences.

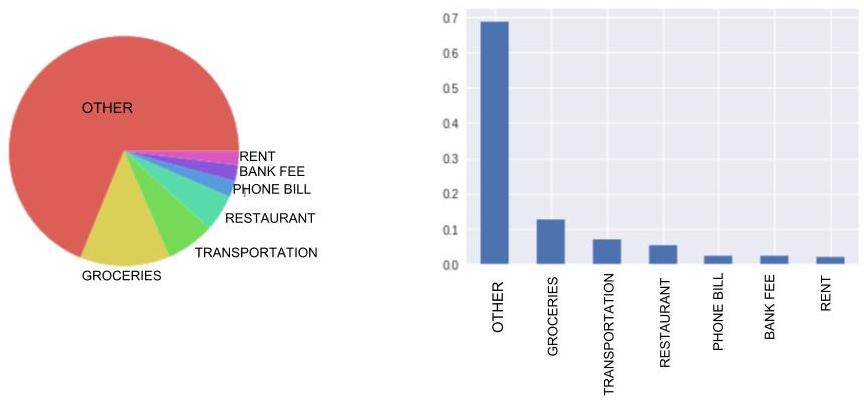
An alternative version of this would be: the empirical distribution of a variable is all of the values (or categories) taken by this variable and their associated frequencies. This can be presented in the form of a table. We will go into this in further detail in the next chapter.

| **category** | **occurence** | **frequency** |
| --- | --- | --- |
| GROCERIES | 39 | 0.126623 |
| OTHER | 212 | 0.688312 |
| TRANSPORT | 21 | 0.068182 |
| ... | ... | ... |

Let’s turn now to graphic representations.

### **Represent Qualitative Variables**

Here are two possible ways of representing the distribution of the *categ* variable:



Two possible ways of graphically representing the "categ" variable

On the left, we have a circular diagram, better known as a pie chart. Here, each section is proportional to the occurrence of the category it represents.

On the right, we have a bar graph. The height of the bars is proportional to the number of occurrences of each category, or (if you prefer) proportional to the frequency of each category, as is the case here.

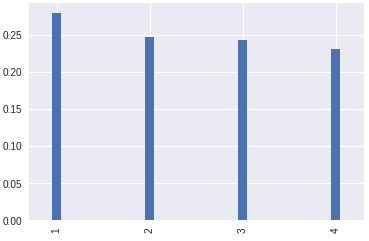
The height of the bars is proportional to the number (or frequency) of banking transactions in a given category, not the sum of their amounts. If you have two transactions in the *transportation* category, one in the amount of $20, and the other in the amount of $300, the height of the bar will be 2 (or 2 / n), not 320.

For ordinal qualitative variables, simply arrange the categories on the graph in increasing order.

### **Representing Quantitative Variables**

#### **Discrete Variables**

Discrete variables are represented by the equivalent of a bar chart: a vertical line graph. Although with qualitative variables, you can place the bars pretty much anywhere along the horizontal axis, with quantitative variables, the lines must be located at precise points. For this reason, it’s best to make your lines very thin.



A vertical line graph

#### **Continuous Variables**

Let’s take the example of a person’s height: this is a continuous variable. It’s quite possible for one person to be 1.47801 meters tall, and another person to be 1.47802 meters tall.

These two heights are different: should we add two lines to our graph to represent them, one for each height?

You’re splitting hairs! 1.47801 meters and 1.47802 meters are practically the same! You should consider these values to be equivalent.

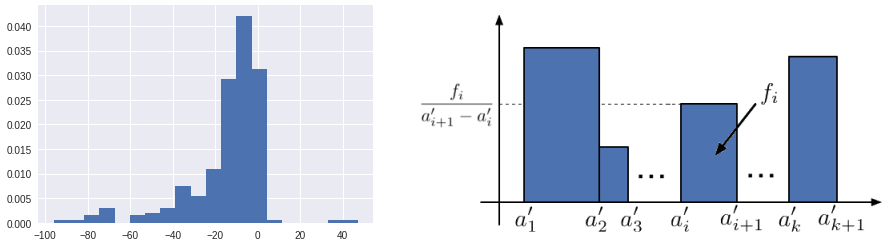
Considering 1.47801 meters and 1.47802 meters to be equivalent is called grouping. We then say that we are **aggregating** these values into **classes** or **bins**. If we decide to group according to height intervals of 0.2 meters, then these two values would both be assigned to the same bin: [1.4m;1.6m[.

Note that it is possible to group values into intervals of unequal widths. For example, we could have intervals of 0.5 meters for heights of under 1 meter, and intervals of 0.2 meters for heights of over 1 meter. It would look like this: *[0m;0.5m[, [0.5m;1m[, [1m;1.2m[, [1.2m;1.4m[, etc.*

Aggregating or grouping variables is called **data binning** (also **discrete binning**, **bucketing**, and **discretization**).

So for continuous variables, we use **histograms** in which the values have been aggregated. Here, because we are representing classes (or bins, if you prefer), we aren’t using thin lines, but instead rectangles whose widths correspond to the size of the bin.

Please note that the occurrence of the category will no longer be represented by the height of the rectangle, but by its *area*. That’s because the bins are not necessarily all the same width.

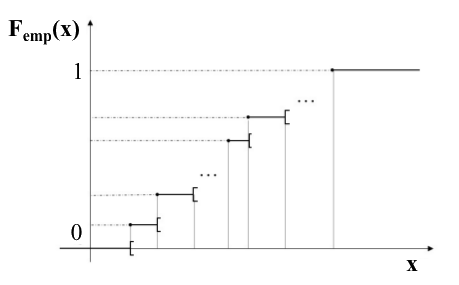


Histogram: width corresponds to the size of the bin

However, if you don’t want to aggregate values, there is another solution: representing the **empirical distribution function**. Think of it as a staircase. To represent it, we place our values along the horizontal axis in increasing order, from small to large. Each time we encounter a value that appears in our sample, we go up a step. So there will be as many steps as there are values—and, moreover, as many individuals. All of the steps are of the same height.

Well... *almost* all of the steps are of the same height. In fact, if there are two equal values or more, then the step is proportionally taller. But for continuous variables, equal values are fairly rare.

When we’ve gone through all of the values of the sample, we will be at the top of the staircase. The staircase is (arbitrarily) assigned a height of 1.



Empirical Distribution Function

To find out more about distribution functions, go to the end of the chapter (*Take It Further*).

### **Now For the Code...**

Here is the code that loads the data from the CSV file :

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

data = pd.read\_csv("operations.csv",parse\_dates=[0])

Here is the code that generated the above graphs. Each graph required only two lines of code:

import matplotlib.pyplot as plt

# QUALITATIVE VARIABLE

# Pie chart

data["categ"].value\_counts(normalize=True).plot(kind='pie')

# This line ensures that the pie chart is circular, not elliptical

plt.axis('equal')

plt.show() # Displays the graph

# Bar graph

data["categ"].value\_counts(normalize=True).plot(kind='bar')

plt.show()

# QUANTITATIVE VARIABLE

# Vertical line graph

data["quart\_month"].value\_counts(normalize=True).plot(kind='bar',width=0.1)

plt.show()

# Histogram

data["amount"].hist(normed=True)

plt.show()

# Prettier Histogram

data[data.amount.abs() < 100]["amount"].hist(density=True,bins=20)

plt.show()

This code is provided in the analysis folder of the downloadable archive in the chapter "Cleanse your Dataset"

Here, we use the same reasoning we used at the beginning of the chapter. We begin by selecting the desired data['categ'] column, then count the number of times each category appears: data['categ'].value\_counts() . To obtain the frequencies, we might want to add normalize=True. This gives us the empirical distribution. To display it, we use the plot method, in which we specify the desired chart type ( pie or bar).

As we said earlier, for a quantitative variable, we group the values into classes (bins), so using value\_counts()doesn’t really make sense. Therefore, we use the hist() method, which groups the values into bins for us. Line 20 creates a histogram that’s a little too spread out, because some amounts are very large and some are very small. For this reason, here we fllter for amounts of between -$100 and $100 using data[data.amount.abs() < 100] (for this we use the absolute value). Finally, we can also specify the number of desired bins using the bins keyword: here, 20.

### **Take It Further: The Empirical Distribution Function**

The Empirical Distribution Function is expressed as follows:

Femp(x)=1n∑i=1nI{xi≤x}

where I is the indicator function.

### **Take It Further: Optimum Number of Data Bins**

With histograms, there are rules for determining the optimum number of bins (classes or intervals) into which a distribution of observations should be grouped. For example, the Sturges rule (1926) considers the optimum number of bins to be:

k=[1+log2(n)]

where n is the sample size.

# **2.4.Represent Variables in the Form of a Table**

There is more to life than histograms!

It is also possible to present variables in the form of a table. It’s not as pretty, of course, but in some cases, this representation is better suited, or supplements, a graphic representation. We will look at four cases, corresponding to the four variable types.

### **Let's Start With Some Vocabulary**

So that the two of us can communicate, you and I, we need to adopt a common language. So we are going to name the different objects we will be manipulating in this chapter.

Here we are working with the bank statement sample, which is made up of transactions. Note that n will represent the number of banking transactions: this is our sample size.

Next, the variable we will be analyzing will be referred to as X .

X is not particularly concrete: it’s just a **variable**. For example, *amount* is a variable.

Our data set contains a number of different values for the amount variable: 1.43, 80, 2.20, etc. No more theory now - all practice! We’ve got real values in front of us. The number of values in our sample is n . So we can express these values as ( x1,...,xn).

When referring to variables, it’s best to use a capital letter. However, when referring to observations of the variable, lower case letters are used. Here, ( x1,...,xn) is an observed realization of the random variable X.

#### **Discrete Quantitative and Qualitative Variables**

If X is qualitative (or even discrete quantitative), it can take a number of categories. For example, *categ* can take the categories “GROCERIES,” “RENT,” “TRANSPORTATION,” etc. We will refer to these categories as {a1,...,ak}, where k indicates the number of categories.

#### **Continuous Quantitative Variables**

To present continuous quantitative variables, we will group the values of variable X into bins, which will be k in number. These bins will be expressed as follows:

{[a1′,a2′[,...,[ak′,ak+1′[}

### **Representing Variables in the Form of a Table**

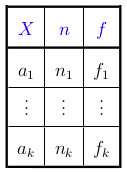
#### **Qualitative Variables**

For qualitative variables, simply count the number of values for each category. This number is referred to as the **occurrence** of the category.

So, for a category ai (where i is between 1 and k of course!), the occurrence is expressed as nI. If we add up the occurrences of all of the categories, we get n: the sample size.

If we divide the number of occurrences by n, we get the **frequency**, which is a number between 0 and 1. As I’m sure you’ve guessed, if we add together the frequencies of all of the categories, we get 1!

Here is how a qualitative variable is normally presented formally, using the example of the *categ* variable:



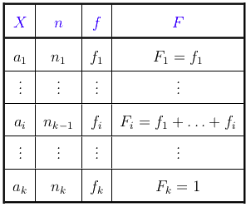
| **categ** | n | f |
| --- | --- | --- |
| OTHER | 212 | 0.688312 |
| GROCERIES | 39 | 0.126623 |
| TRANSPORTATION | 21 | 0.068182 |
| RESTAURANT | 16 | 0.051948 |
| PHONE BILL | 7 | 0.022727 |
| BANK FEE | 7 | 0.022727 |
| RENT | 6 | 0.019481 |

This example works for a qualitative variable. If the variable is ordinal qualitative, simply arrange {a1,...,ak} in increasing order.

#### **Quantitative Variables**

##### **Discrete Variables**

For discrete quantitative variables, we can take the preceding table and add a column to it providing the **cumulative frequency**. The cumulative frequency of a category ai is simply the sum of the frequencies of all of the categories that are less than or equal to ai. It is expressed as F. Here is an example using the *quart\_month* variable:

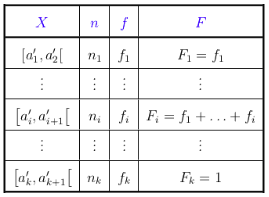


| quart\_month | n | f | F |
| --- | --- | --- | --- |
| 1 | 86 | 0.279221 | 0.279221 |
| 2 | 76 | 0.246753 | 0.525974 |
| 3 | 75 | 0.243506 | 0.769481 |
| 4 | 71 | 0.230519 | 1.000000 |

The cumulative frequency of the final category (ak ) is 1.

##### **Continuous Variables**

For continuous variables, simply replace {a1,...,ak} with bins, as we saw earlier. Here’s what that would look like, using the *amount* variable:



| amount | n | f | F |
| --- | --- | --- | --- |
| [...] | [...] | [...] | [...] |
| [-120.0, -90.0[ | 2 | 0.006494 | 0.048701 |
| [-90.0, -60.0[ | 11 | 0.035714 | 0.084416 |
| [-60.0, -30.0[ | 28 | 0.090909 | 0.175325 |
| [-30.0, 0.0[ | 237 | 0.769481 | 0.944805 |
| [0.0, 30.0[ | 3 | 0.009740 | 0.954545 |
| [...] | [...] | [...] | [...] |

### **Now for the code...**

In Python, the code is pretty simple. All you need (almost) is one line of code per column. Here is the code that generated the summary table for the *quart\_month* variable.

occurrences = data["quart\_month"].value\_counts()

categories = occurrences.index # the occurrences index contains the categories

tab = pd.DataFrame(categories, columns = ["quart\_month"]) # creation of table based on categories

tab["n"] = occurrences.values

tab["f"] = tab["n"] / len(data) # len(data) returns the sample size

To calculate the occurrence, we use value\_counts() for the variable we want to look at. This method returns a *Series object* whose values are occurrences, and whose index contains the categories (lines 1 and 2).

Need a little refresher on Series objects and their indices? Go [**here**](https://openclassrooms.com/fr/courses/2304731-learn-python-basics-for-data-analysis). :ange:

Based on our categories, we create the tab table (line 4), to which we add an occurrence column (line 5) and then a frequencies column (line 6).

To calculate cumulative frequencies, you need only 2 lines more. One line sorts the values, the other calculates the sum of the cumulative frequencies:

tab = tab.sort\_values("quart\_month") # sorts values of variable X (increasing)

tab["F"] = tab["f"].cumsum() # cumsum calculates the cumulative sum

**2.5.QUIZ**

**Question 1**

**A qualitative variable can be:**

either categorical or ordinal

**Question 2**

The variable "Juice Size" has the following categories: “small,” “medium,” and “large". This variable is, therefore:

Ordinal

**Question 3**

| **Color** | **Occurence** | **Frequency** | **Percentage** |
| --- | --- | --- | --- |
| Blue | 39 | 0.27857 | 27.9 |
| Red | 56 | 0.40 | 40 |
| Green | 21 |  |  |
| Yellow | 24 | 0.1714 | 17.1 |

A candy company claims that for any given box of their candy, there is **never fewer than** 13% of each color. Being the scientist you are, you decide to investigate their claim.

What is the frequency and percentage of the color Green?

0.15 (15%)

To find the frequency and percentage of the color Green:

### Given data:

* Total occurrence of all colors = 39+56+21+24=14039 + 56 + 21 + 24 = 14039+56+21+24=140

### Frequency of Green:

21140=0.15\frac{21}{140} = 0.1514021​=0.15

### Percentage of Green:

0.15×100=15%0.15 \times 100 = 15\%0.15×100=15%

### ✅ Answer:

The frequency and percentage of the color Green are 0.15 (15%).

**Question 4**

**Before calculating a cumulative frequency, what precaution must you take?**

Order the categories (or aggregation bins) in increasing (or decreasing) order.

**Question 5**

**Which of the following sentences is FALSE?**

An individual contains more than one sample.

**Question 6**

**Which of the following sentences is FALSE?**

The bars of a histogram are always all of the same width

**Question 7**

Which sentence(s) is/are TRUE about data binning?

Data binning is a way to aggregatevariables

Bins can have unequal widths

Bins are used for continuous variables

All of the above

**Question 8**

Which of the following are continuous variables?

Careful, there are several correct answers.

Age

Volume of water in a pool

# **3.1.Understand the Purpose of Univariate Analysis**

### **What is a Statistic?**

What do the following have in common: the average age of the population of China in 2010, the pass rate for the quiz at the end of Part 1 of this course, and the soil erosion index for the Hauts de France region in France?

Answer: They are all **statistics**!

Formally, a **statistic** is a numerical indicator calculated on the basis of a sample. The average age is calculated on the basis of the residents of the country, the pass rate of a quiz is calculated on the basis of student responses, and the soil erosion index is calculated on the basis of samples taken from plots of land.

In other words, as soon as you calculate a number based on a sample, you’re generating **statistics**!

Statistics are useful, because they enable us to reduce characteristics of a large sample to a single number! Of course, a lot of information gets lost in the calculation; for example, you can calculate a pass rate based on student responses, but you can’t find student responses from a pass rate alone!

So, a statistic is an indicator (of lesser or greater utility) of a specific characteristic of a sample.

The number of statistics being generated all over the world in every field is such that the term has spun off numerous synonyms: **statistical data**, **statistical indicators**, **statistical measures**, etc.

You might also run across the term **statistical index**. A statistical index is a statistical measure developed according to a certain vision, based on specialized knowledge in a specific field (the economy, for example). You could say that an index is a statistic embedded in a philosophy. Unlike an index, an **indicator** (such as an average, for example) is entirely neutral.

In the chapters that follow, we will look only at certain statistical *indicators* (or measures), not at *indices*, because indices require outside knowledge in a specific field.

### **What are Statistical Indicators and Indices Used For?**

The reason we develop so many indices and indicators is that, when we’re trying to make decisions, they can point us in the right direction (as their name suggests!). For example, economic, environmental, and sociological indicators are used to help make political decisions.

Some indicators and indices are developed on the basis of very simple calculations, such as a company’s sales figures (which are obtained simply by adding up all of the company’s receipts).

However, others require more complex calculations, such as those involving more than one characteristic of a population. This is the case for the **Human Development Index (HDI)**, which is developed on the basis of per capita GDP, life expectancy at birth, and education level. There is also a [relational capacity indicator](http://www.ferdi.fr/en/node/879), which measures the quality of relationships between people and their level of relational autonomy.

In the environmental field, we find indices of "biocapacity" and “human environmental footprint,” which are calculated on the basis of data concerning forests, developed land, agricultural fields, etc.

### **What do Indicators Tell Us?**

Of course, many different indicators can be developed for a single population. Each of them tells us something about a specific characteristic of that population. For example, the class average on an exam tells us whether the exam had a good pass rate. But the standard deviation (more on this soon) of the grades of this same population tells us whether there were major discrepancies between students.

You should never trust an indicator 100%. I’m sure you realize that when you reduce a complex reality to a single number you are necessarily overlooking important aspects of that reality. So when you use an indicator, you must always understand what it is measuring—and what it is *not* measuring!

# **3.2.Calculate Measures of Central Tendency**

In this section, we are going to perform **univariate analyses**. Univariate analyses focus on only **one** variable at a time.

Say you need to drive to a job interview located some distance away in another city, and you wonder what time you should leave in order to arrive there by 3:00 p.m. Since you have a lot to do in the morning, you don’t want to leave too early, but you still want to make sure you get there on time.

You aren’t familiar with the route you’ll be taking but, fortunately, a friend of yours drives it every day. He knows it by heart.

"How much time does it take to drive from this city to that city?" you ask.

"It depends on how much traffic there is. Most of the time it takes me 40 to 45 minutes," he answers.

Let’s look at this sentence. Your friend is familiar with the route; he’s driven it perhaps 1,000 times! Each time, he took note (more or less unconsciously) of how long it took. So we have a sample of 1,000, with one *continuous quantitative variable*: the time it took to drive between the two cities.

Even though, in theory, the trip time can take values of between 0 and infinity, I’m sure you know that these trip times *concentrate* around a certain value. What you want here is some idea of where (on an axis from 0 to infinity) the trip time values *concentrate*.

.” The word concentrate contains the word *centre*, does it not?

Exactly! We have now arrived at the topic of this chapter: **measures of central tendency.** We’ll look at three such measures, and—guess what!—they all start with the letter M!

### **Measures of Central Tendency**

#### **The Mode**

Most of the time, it takes me 40 to 45 minutes.

When your friend said that, he was giving you a measure of central tendency called the **Mode**.

For qualitative variables, or for discrete quantitative variables, the mode is the most commonly occurring category or value. In our bank statement, the mode of the *categ* variable is “Other,” because the “Other” category occurs 212 times in the sample, and all of the other categories (“rent,” “groceries,” etc.) occur fewer times.

For continuous quantitative variables, we aggregate values, grouping them into bins (intervals). The **modal class** interval is the one that has the greatest frequency. Your friend divided his variable into intervals of 5 minutes, and determined that the most frequently occurring interval was *[40min;45min[*.

#### **The Mean**

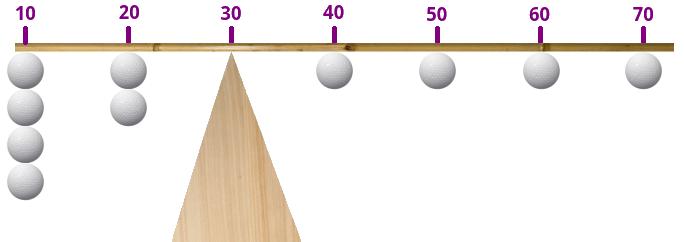
To your friend’s statement, you reply: "Yes, but I can’t be satisfied with knowing just the most frequent duration of the trip, because if the second-most frequent duration is 65 to 70 minutes, I’ll need to leave a lot earlier!"

"You’re right. In fact, I average 60 minutes per trip, because there’s often traffic," he replies.

This changes everything! It’s a good thing you asked him to be more precise; otherwise, you might have arrived late! Here, your friend is speaking in terms of the **Mean**.

You all know what the Mean is. To calculate the Mean, you add up all of your values and divide them by the number of values you added together. It is common to associate the Mean with the concept of balance and center of gravity. Why? Imagine that you have 10 numerical values. You take a pole and graduate it, marking the position of each of your 10 values with a pen. Then you attach balls to each of the ten marks. You calculate the Mean of your ten values, and mark that on the pole. Then, if you want to balance this pole horizontally, you have to find its center of gravity. You see where I’m going with this, right? The center of gravity will be exactly where you marked the mean of your ten values!

We are assuming that the weight of the pole is negligible in comparison to that of the balls, which all weigh the same.



The center of gravity is located at 30

Yes, but on our pole... if one of the values is very different from the others, its ball will be very far from the others on the pole, and the pole will be totally off balance, all because of a single value!

#### **The Median**

As you may have guessed, the problem of the unbalanced pole brings us to the issue of outliers. As we just saw, the Mean is *very sensitive* to outliers.

So, with regard to your trip, you ask your friend:

When you told me it takes you an average of 60 minutes, your calculations probably included the rare times when there was snow on the ground and it took you four hours to make the trip. These are outliers, and don’t need to be included, because it’s summer, so there won’t be any snow!

To which your friend replies:

You’re right. So let me put it another way: let’s say that half of the trips I have made took over 55 minutes, and the other half took less than 55 minutes.

Your friend is now speaking in terms of the **Median**.

The Median, (abbreviated *Med*), is the value above and below which the number of observations is the same.

To find the median of your n values, you can begin by sorting them. Once they are sorted, you call the first value x(1) , the second value x(2), ... , and the final value x(n). The median is the value that is right in the middle, or Med=x(n+1/2).

So, out of n=999 trips, the median is x(500)=55minutes.

Your calculation works because 999 is an odd number. But if there are 1,000 trips, is the median the 500th value or the 501st value? If you choose the 500th, there are 499 values below it and 500 values above it. But if you choose the 501st, there are 500 values below it and 499 values above it: in both cases, you’re off balance!

You’re right. In this case, you split the difference: place the median at the center between the 500th and 501st value. Thus, if there are 1,000 trips, and the 500th value is x(500) = 54 minutes 30 seconds, and the 501st value is x(501) = 55 minutes 30 seconds, we split this in half and get 55 minutes.

More formally, if n is even, the median is Med=1/2(x(n2)+x(n2+1))

### **Now For The Code...**

It would be hard to get any simpler than the code for this: there is only one line of code per indicator! Let’s take the example of the *amount* variable in our account statements:

data['amount'].mean()

data['amount'].median()

data['amount'].mode()

Each of these lines returns a value except for line 3, which returns pd.Series, because a distribution can have more than one mode (see the *Take It Further* section).

The transaction amounts vary widely: there are expenditures (negative amounts) that are sometimes quite large (rents, for example) and expenditures that are often small (groceries, phone, etc.), and there is money coming in (positive amounts) less frequently but in large amounts. It is therefore difficult to interpret the mean (which is very sensitive to atypical values). Here the mean is **$2.87**. We have the same problem with the median, which is **-$9.60**. The fact that it is negative tells us, however, that there are more debits than credits. On the other hand, the mode tell us that most of the transactions are around **-$1.60**. Here, the three measures are very far apart.

To get more uniform transaction amounts, I suggest you calculate these three measures for *each* transaction category. Within a category, the amounts will be more similar, since the transactions will all be of the same type.

So I suggest a “for” loop that will iterate over each of the categories:

for cat in data["categ"].unique():

subset = data[data.categ == cat] # Creation of sub-sample

print("-"\*20)

print(cat)

print("mean”:\n",subset['amount'].mean())

print("med:\n",subset['amount'].median())

print("mod:\n",subset['amount'].mode())

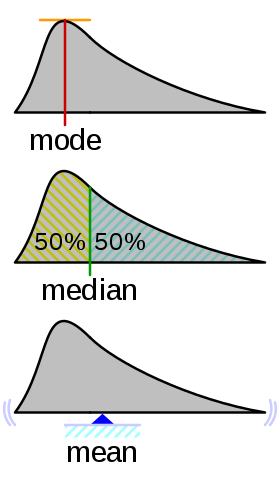
subset["amount"].hist() # Creates the histogram

plt.show() # Displays the histogram

For each category, we create a sub-sample (**subset**) containing only the transactions of the current category. On lines 5 to 7 we display the three measures, and also the histogram, so that we can view all three measures in perspective. I leave it to you to interpret your results!

### **Take It Further: Using a Histogram**

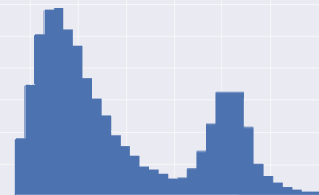
On a histogram, the Mode is the “highest point” of the distribution, the Median is the value that divides the area in two, and the Mean is the center of gravity of the distribution, as shown in this illustration:



Source: commons.wikimedia.org, licence GFDL

### **Take It Further: Multimodal Distributions**

More generally, we sometimes extrapolate the Mode by equating it to the peak(s) of a distribution. Modes do not necessarily occur singly. When a distribution has only one peak, we speak of a unimodal distribution. But a distribution may also have two or more peaks, in which case we speak of a bimodal or multimodal distribution. Finding the number of modes of a distribution is of interest in statistics where there are multimodal distributions.



Bimodal Distribution

# **3.3.Calculate the Value Spread in your Data**

This chapter contains mathematical formulas, but don’t let that scare you! They will be explained step-by-step.

In the last chapter, your friend gave you an estimate of how long your trip should take. But he gave you measures of central tendency, such as, for example, the Mean, which in this case was 60 minutes per trip.

What you need to know now is whether the durations of your friend’s trips “tightly clustered” around 60 minutes (for example: [58, 60, 62, 59, 57...]), or whether they were more loosely dispersed (example: [40, 70, 78, 43, .etc.]).

Why?

If the values tightly cluster around 60 minutes, then you want to leave 75 minutes in advance. That way, you will be likely to arrive 5 or 10 minutes before your interview. But if the values are very far apart, you should leave 100 minutes in advance of your interview, because it’s entirely possible for the trip to take 80 minutes!

I get it! Calculating the value spread... I’m guessing there’s a statistical measure for that?

Absolutely! There are even several. They are called **measures of dispersion**.

### **Thinking It Through...**

Let’s try to create our own indicator of dispersion, step-by-step. To illustrate it, we’ll use the following values (70, 60, 50, 55, 55, 65, 65), and give each of them a name: Xi , with i taking values from 1 to 7. So our value names will run from x1 to X7 .

Formally, this is expressed: (x1,...,xn)=(70,60,50,55,55,65,65) , with n=7 .

Note that the mean of these values is 60, which is written x¯=60 , and pronounced “*x bar*.”

Measures of dispersion are easy! Let’s take all of our values and calculate the deviation from the mean for each of them. Then we will add up all of the deviations.

That’s a good start. Because our mean is 60, the deviations of xi from the mean are: (x1−x¯,...,x7−x¯)=(10,0,−10,−5,−5,5,5). Except, if we add them all up, we get 0! We can demonstrate this mathematically: no matter how dispersed your values, the sum of the deviations from the mean will always be 0. Not very helpful...

It’s 0 only because there are positive and negative numbers. Let’s get around that by squaring them all. A square number is always positive isn’t it?

That’s right! When we do that, here’s what we get: ((x1−x¯)2,...,(x7−x¯)2)=(100,0,100,25,25,25,25). Now if we add up all of these values, we get 300.

Here, we calculated the sum of all of the (xi−x¯)2 with i taking values from 1 to 7. Mathematically, it’s expressed like this:

∑i=1i=7(xi−x¯)2

Okay. But there’s another problem. Here we have 7 values, simply because we are a little bit lazy and only collected 7. But in statistics, the more we collect, the better an idea we have of what we are describing. So we should have collected 10, 100 or even 1,000 values!

However, with 1,000 values, our measure would explode! It would go from 300, with 7 values, to perhaps 40,000,000,000, with 1,000 values. That’s a problem.

So, instead of calculating the sum and blowing up our indicator, let’s take the mean. Whether there are 7 values or 1,000, the mean will not explode.

Good idea. The mean of (100,0,100,25,25,25,25) is 42.86.

To obtain the mean of 42.86, we simply divided 300 by the number of values: 7. We multiplied the previous formula by

1n , which gave us 1n∑i=1i=n(xi−x¯)2

### **Measures of Dispersion**

#### **Empirical Variance**

Guess what! The indicator we just made is one of the mostly used common indicators in statistics! It’s called the Empirical Variance.

As we just saw, it is equal to v=1n∑i=1n(xi−x¯)2

To go further into the calculations aspect, go to the *Take It Further* section at the end of the chapter. You will also see a “corrected” version of the Empirical Variance referred to as unbiased. For this I also refer you to the *Take It Further* section. :magicien:

#### **Standard Deviation**

The standard deviation is simply the square root of the variance. It is often abbreviated *std*. In fact, when we calculate the empirical variance of our trip times, the result is expressed in a unit of minutes 2 , which is not very intelligible. If we take the square root, the unit becomes the minute. Here, our standard deviation is 6.55 minutes. It is written √s=v

But a standard deviation of 6.55 minutes for a trip of 1 hour (on average) is not the same as a standard deviation of 6.55 minutes for a trip of 24 hours (on average)! This is called a **coefficient of variation**, which you will find in the *Take It Further* section.

#### **Inter-Quartile Range**

Remember the Median? It’s the “middle” value dividing the higher and lower values.

A **quartile** is the same thing, except the values are divided by quarters. So there are three quartiles, written Q1 (first quartile),

Q2 (second quartile) and Q3 (third quartile). It works like this:

• 1/4 of the values are below Q1 , 3/4 are above.

• 2/4 of the values are below Q2 , 2/4 are above ( Q2 is the Median!).

• 3/4 the values are below Q3 , 1/4 are above.

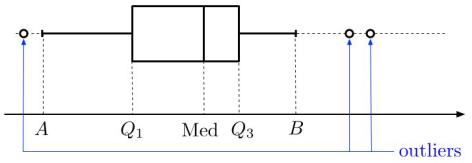
The general concept behind this is known as the **quantile of order alpha** α . Thus, the Median is the quantile of order 0.5; Q1 is the quantile of order 0.25; and Q3 is the quantile of order 0.75. There are also **deciles** (quantiles of order 0.1, 0.2, etc.), and **centiles**, which are also called **percentiles** (quantiles of order 0.01, 0.02, etc.).

The Inter-Quartile Range is the difference between the 3rd quartile and the 1st quartile:IQ=Q3−Q1

#### **Box-and-Whisker Plots (Box plots)**

Box and Whisker: what a funny name! French people call it a “moustache box.”

A box plot is a way of graphically representing groups of numerical data through their quartiles, to indicate the degree of dispersion in the data. Q1 and Q3 determine the sides of the box, and the median is often indicated inside. Lines (the “whiskers”) then extend vertically from the box to indicate the lowest and highest values...provided that these lines (on either side of the box) are within 1.5 IQR of the upper and lower quartiles of the inter-quartile range. Values below Q3−1.5IQ or above Q3+1.5IQ are considered outliers; they are not included in the whisker:



Box and Whisker Plot

Box plots and histograms have the same purpose: to give an indication of the empirical distribution. Box plots may seem more primitive than histograms, but they have the advantage of making it easier to compare the distribution of more than one variable (two box plots are easier to compare than two histograms).

### **Now for the code...**

For the empirical variance and the standard deviation, the same principle applies as in the last chapter: we call the var() and std() methods for the variable we are looking at. Full disclosure: these two methods will return results that vary somewhat from those you get with the above data formulas. This gets us into “biased estimators” (for this, the more motivated among us can refer to the section *Take It Further: Empirical Variance*). To obtain the results described above, we need to add ddof=0 (lines 8 and 9).

We will display the box plots with the histograms, so you can compare :). On line 12, the keyword vert=False is there because we don’t want the box plot to be vertical (we want it to be horizontal!). Taking the code from our previous chapter, here’s what we get:

for cat in data["categ"].unique():

subset = data[data.categ == cat] # Creation of sub-sample

print("-"\*20)

print(cat)

print("mean:\n",subset['amount'].mean())

print("med:\n",subset['amount'].median())

print("mod:\n",subset['amount'].mode())

print("var:\n",subset['amount'].var(ddof=0))

print("ect:\n",subset['amount'].std(ddof=0))

subset["amount"].hist() # Creates the histogram

plt.show() # Displays the histogram

subset.boxplot(column="amount", vert=False)

plt.show()

The variance of the transaction amount for the RENT category is null, meaning the rent amount is always the same.

### **Take It Further: The Corrected Empirical Variance**

The best way of estimating the variance of a random variable (i.e., the theoretical variance) is not the Empirical Variance.

Surprising, no? Yes. When we get deep into the calculations, we see that the empirical variance gives values that are lower (on average) than the variance of the random variable.

This gets us into the concept of estimator bias. An *unbiased estimator* is better than a biased estimator, and empirical variance is a biased estimator of the variance of a random variable.

The corrected—or unbiased—empirical variance was developed to correct this bias. It is often expressed as s′2 , and is equal to = s′2=nn−1v , where v is the empirical variance, and n is the sample size. When the sample size is large, the empirical variance and the unbiased empirical variance are almost equal.

### **Take It Further: Calculations Using the Empirical Variance**

We can show by calculation that the Empirical Variance v can also be expressed in a very practical form:

v=1n∑i=1n(xi−x¯)2=(1n∑i=1nxi2)−x¯2

This is the Koenig-Huygens theorem. For a demonstration, go [here](https://en.wikipedia.org/wiki/Algebraic_formula_for_the_variance).

If we create a new variable Y based on a variable X whose variance vX is known, and Y=aX+b, we can find the variance of Y, which is written vY! It is given by this relation: vY=a2vX .

### **Take It Further: The Variation Coefficient**

When you travel, a standard deviation of 6.55 minutes for a trip of 1 hour is not the same as a standard deviation of 6.55 minutes for a trip of 24 hours! In the first case, the standard deviation will be seen as fairly large, while in the second case, it will be considered negligible over a 24-hour period.

The variation coefficient was developed to address this. It is the standard deviation divided by the mean:

CV=sx⎯⎯⎯

### **Take It Further: Other Measures of Dispersion**

At the beginning of this chapter, we said:

*Let’s square all of the numbers—a square number is always positive, right?*

When we said this, perhaps you thought:

Couldn’t we also take the absolute value instead of squaring the number?

Absolutely. ;) When we do this, we are calculating the **Mean Absolute Deviation**.

There are two versions: one in which we calculate the deviations from the mean, the other in which we calculate the deviations from the median.

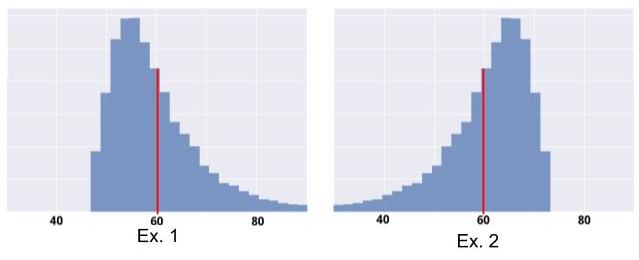
Here is the version with the median: EMA=1n∑i=1n|xi−Med|

If you want a more robust calculation, we can also define the **Median Absolute Deviation**, which is the median of the absolute deviations from the data’s median.

# **3.4.Calculate Measures of Shape**

Okay, so your friend gave you the average time of his trips, as well as the standard deviation. You can start to relax a little. But...here’s something you haven’t thought about.

Look at these two distributions:



They have the same empirical mean (60 minutes), and the same standard deviation. However, Example 1 is “riskier” than Example 2. That’s right, in Example 2, it is highly unlikely that your trip will take more than 75 minutes: no risk of being late! But in Example 1, your trip could very well take 80 minutes, or even longer.

Notice that it’s not enough to know the average and the standard deviation. What you need to know here is the shape of the distribution: does it skew more to the left or more to the right?

There are statistical measures for that! They are called **Measures of Shape.**

### **Thinking It Through...**

Let’s make our own shape indicator! We want to know if the distribution skews more to the left of the mean, or more to the right of it.

This is equivalent to knowing whether the majority of the values are above the mean, or below the mean.

I suggest you take what we built in the last chapter. First, we had this idea: Let’s take all of our values, and calculate the deviation from the mean for each one. Then let’s add these deviations together!

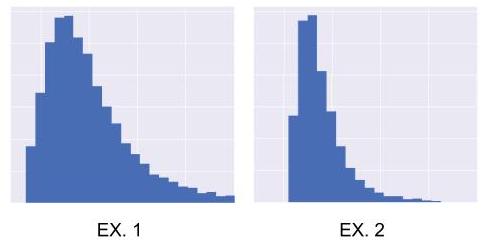
We expressed the deviation of a value from the mean as (xi−x¯). If this deviation is positive, it means that xi is above the mean; if it is negative, it means that xi is below the mean.

When we added all of these deviations together, we noticed that the result was always 0. So we squared the number: (xi−x¯)2 . With a squared number, the result is always positive. However, if it is always positive, we lose the information that tells us whether xi is above or below the mean. And here, we want to keep that information!

Okay, if squaring doesn’t work, what happens if we cube the results?

Good idea! When we cube the deviation, we obtain (xi−x¯)3 . Unlike the squared number, the cubed number retains the negative sign of (xi−x¯). Next, we take the average of all of these cubed deviations, and obtain: 1n∑i=1n(xi−x¯)3 .

We have achieved our goal: the amount will be negative if most of the values are below the mean; otherwise it will be positive!

But we can do even better. Take a look at these two distributions:

They have the same shape, but not the same standard deviation (distribution 1 is more spread out than distribution 2; distribution 1 has a standard deviation that is twice that of 2). Because they have the same shape, we would like for our indicator to have the same value for both distributions.

But currently, that’s not the case. In Example 1, the deviations from the mean are twice as great as in Example 2. Because we are cubing these deviations, our indicator will be 23 greater for 1 than for 2. But we want them to be equal. So to correct this, we need to nullify the effect of the standard deviation. To do this, we are going to divide our indicator by the cubed standard deviation:

1n∑i=1n(xi−x¯)3s3

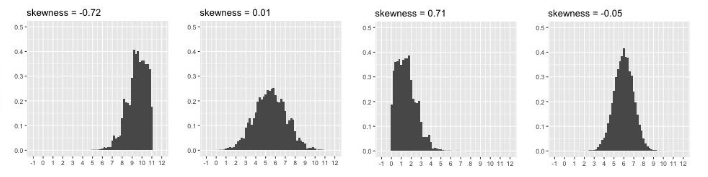
### **Measures of Shape**

#### **Skewness**

Guess what! The indicator we just created is commonly used by statisticians: it’s called **Empirical Skewness**. In general, skewness γ1, and its numerator μ3 , are expressed: γ1=μ3s3

with 1n∑i=1n(xi−x¯)3 Skewness is a measure of asymmetry (or symmetry). The asymmetry of a distribution is the regularity (or lack thereof) with which the observations are distributed around a central value. It is interpreted as follows:

* • If γ1=0, the distribution is perfectly symmetrical.
* • If γ1>0, the distribution is positively skewed, or skewed right.
* • If γ1<0, the distribution is negatively skewed, or skewed left.

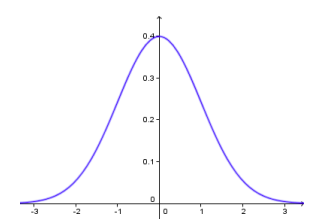


Relation Between Shape of Distribution and Skewness

For more on the concept of **asymmetry**, go to the *Take It Further* section at the end of this chapter. You will see how comparing the median to the mean can also be used to measure asymmetry.

#### **Empirical Kurtosis**

**Empirical Kurtosis** is not a measure of asymmetry; it’s a measure of “peakedness,” or “flatness.” **Peakedness** can be interpreted whenever the distribution is symmetrical. It is determined in relation to the most famous distribution of all: the so-called **Normal Distribution** (also known as the Gauss, Gaussian, or Bell Curve). I’m sure you’ve seen it. It looks like this:

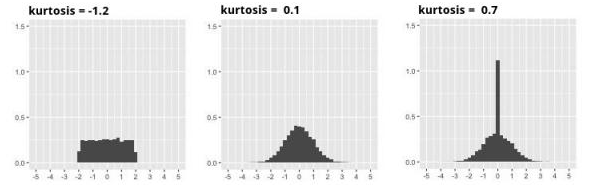


The Normal Distribution Kurtosis is often expressed γ2, and calculated by:γ2=μ4s4−3 with μ4=1n∑i=1n(xi−x¯)4

What are those mysterious μ3 and μ4 notations in the skewness and Kurtosis formulas, you ask? These are **Moments**. For more on these, see the *Take It Further* section at the end of the chapter. :D

Kurtosis is interpreted as follows:

* If γ2=0, the distribution has the same degree of peakedness (or flatness) as the normal distribution.
* If γ2>0, the distribution is more peaked (less flat) than the normal distribution: the observations are more densely concentrated.
* If γ2<0, the distribution less peaked (flatter) than the normal distribution: the observations are less densely concentrated.



Relationship Between Shape of Distribution and Kurtosis

### **Now for the code...**

You know how it works! Take the code from the previous chapter and add some lines: here, 10 and 11, to calculate Skewness and Kurtosis:

for cat in data["categ"].unique():

subset = data[data.categ == cat] # Creation of sub-sample

print("-"\*20)

print(cat)

print("mean:\n",subset['amount'].mean())

print("med:\n",subset['amount'].median())

print("mod:\n",subset['amount'].mode())

print("var:\n",subset['amount'].var(ddof=0))

print("ect:\n",subset['amount'].std(ddof=0))

print("skw:\n",subset['amount'].skew())

print("kur:\n",subset['amount'].kurtosis())

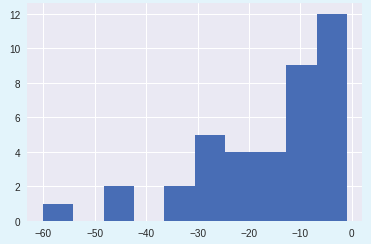
subset["amount"].hist() # Creates the histogram

plt.show() # Displays the histogram

subset.boxplot(column="amount", vert=False)

plt.show()

The skewness of all of the transactions whose categories mainly concern expenses (phone, groceries, etc.) is very likely to be negative. The distribution of expenses is skewed left, because there are frequent small purchases and, less frequently, very large purchases (hence, skewed far left):



Empirical Distribution of Expense Amounts in the GROCERIES Category

### **Take It Further: A Word About Asymmetry**

Remember this sentence from the beginning of the chapter?

“We want to see whether the majority of the values are above the mean or below the mean.”

When we say *majority*, we mean over 50% *of the values.* You will recall that the median is the middle value: 50% of the values are above it. Therefore, the above sentence is equivalent to saying: *We want to know if the median is greater than or less than the mean.*

To find out which is greater, the median or the mean, we analyze the **asymmetry** of the distribution!

A distribution is considered symmetrical if its shape is the same on either side of the center of the distribution. In this case: Mode=Med=x¯

A distribution is *skewed right* (or is *positively skewed*, or has *positive asymmetry*) if:Mode<Med<x¯.

Similarly, it is *skewed left* (or is *negatively skewed*) if: Mode>Med>x¯.

### **Take It Further: Moments**

The Empirical Mean, Empirical Variance, μ3 and μ4 are all **Moments.**

The *Mean*, *Variance*, *and Measures of Shape* we have seen characterize the *geometry* of the distribution, which is what makes it similar to the definition of **Moment of Inertia**.

Indeed, people who study mechanics often calculate moments. For example, if you take a graduated ruler and attach a weight to each spot that corresponds to an observation (x1,...,xn) , and then rotate this ruler about the mean value, the moment of inertia will be calculated the same way as the variance of (x1,...,xn)!

In statistics, the **general empirical moment** of order p in relation to t is given by the relation:

Mtp=1n∑i=1n(xiat)p

The **simple empirical moment** is the general moment about t=0 : Mp=1n∑i=1nxip

The **central empirical moment** is the general moment about the mean, or t=x¯: μp=1n∑i=1n(xi−x¯)p

Here again we see the general formula of our μ3 and μ4 , and we also see that μ2 is the empirical variance, and that μ3=0 (the sum of the deviations from the mean is always zero, as we saw in the last chapter). Finally, the mean is the simple moment of order 1: M1=x¯.

# **3.5.Explore Measures of Concentration**

Good news: no more hiring interview and no more friend who, instead of just telling you how much time you need to allow in order to arrive at your interview destination on time, talks about medians, means, variances, skewness, and all that jazz!

Let’s get back to your bank statements and analyze your expenses.

An expense is an amount of money. That’s good, because **Measures of Concentration** are most often used with sums of money! When you analyze a concentration of money, you look at how evenly distributed it is (or is not).

We are going to look at whether all of the money you spend is *concentrated i*n a few banking transactions, or whether, instead, it is evenly distributed across all of your transactions. Your spending will be considered “concentrated” if you generally make a lot of small purchases, but from time to time make an enormous one. It will be considered “evenly distributed” if, on the other hand, the amounts of your (outgoing) banking transactions tend to be approximately the same. To visualize this, we will use the **Lorenz Curve**.

### **Measures of Concentration**

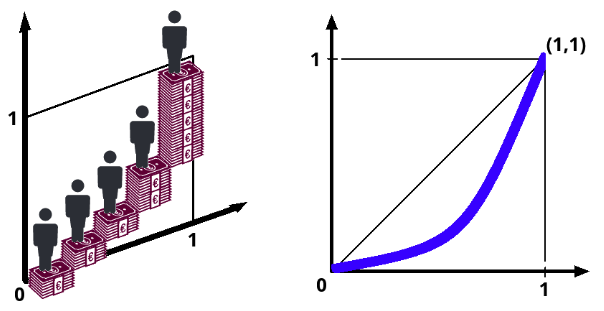
#### **The Lorenz Curve**

To get an idea of the Lorenz Curve, imagine the population of a country and focus on companies who have income: companies who are making money. Think of the **Lorenz Curve** as a podium, only not with just 3 steps, but with as many steps as there are companies. The podium resembles a staircase. The company who makes the most money is at the top, and the company who makes the least money is at the bottom.

Except, this staircase is uneven: the height of a given step in relation to the step before it corresponds to the income of the company who is standing on it. So the step of a company who makes a lot of money will be very tall in relation to the step that precedes it.

Question: what is the total height of the staircase?

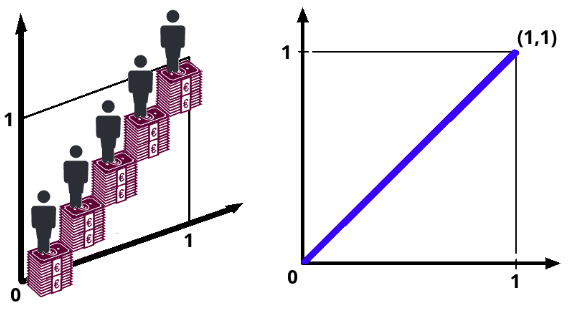
The height of the staircase is equal to the *sum of the heights of the steps*. The sum of the step heights is equal to the sum of the individual incomes. For example, if $10,000 has been distributed among the population, the height of the staircase will be 10 meters (assuming that one meter represents $1,000). The Lorenz Curve graphically represents this staircase, except that the height of the staircase is assigned a value of 1, as is the length of the staircase (projected across the bottom).



The Lorenz Curve

What happens if every company has the same amount of money?

In this case, the income distribution would be perfectly equal, and the staircase would look like the one on the left below:

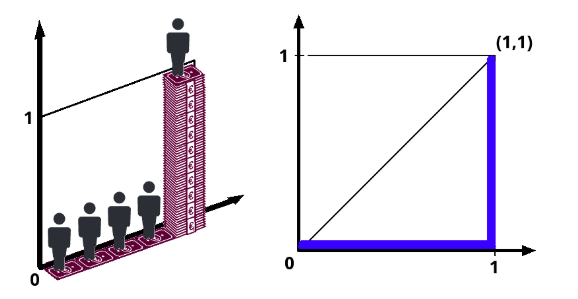


The heights of the individuals steps are the same

As you can see, the heights of the individual steps are exactly the same, and the people on them line up in a 45-degree angle called the *line of perfect inequality*, a line that passes through points (0.0) and (1.1). In the graph on the right, the line is represented in blue.

What if all of the wealth is concentrated in the hands of just one company?

This is the opposite extreme of the previous one. Here, the distribution is as unequal as possible:



Unequal distribution

Here, the Lorenz Curve does not at all aligned with the first bisector. It diverges as much as possible from it!

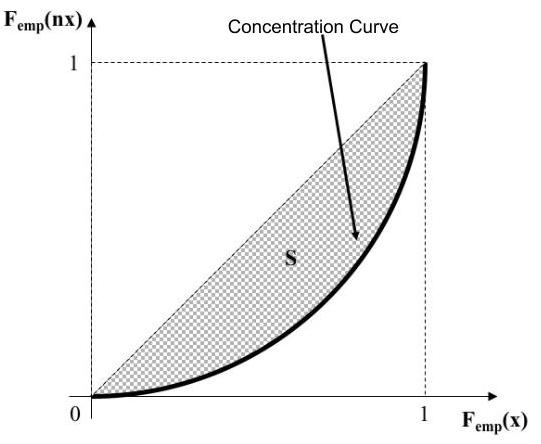
In short, the further the Lorenz Curve diverges from the first bisector, the greater the inequality in wealth distribution.

#### **The Gini Index**

The Lorenz Curve is not a statistic; it’s a curve! Therefore, the **Gini Index** was developed to interpret the Lorenz Curve.

The Gini Index measures the area between the Lorenz Curve and the first bisector. To be precise, if this area is expressed as S, then

gini=2×S.



The Gini Index

The more concentrated the income distribution, the higher the percentage of income held by the richer segment of the population, and the lower the percentage held by the poorer segment of the population. In a perfectly equal distribution, 10% of the population would receive 10% of the overall population’s total income.

#### **Other Ways of Expressing Concentration**

Were they to hear about the Gini Index in the media, the general public would not find it very meaningful. A more intelligible way of expressing inequality is:

* X% of the population owns Y% of the world’s wealth, or
* X% of top-income earners own as much as Y% of low-income earners.

The first of these formulations relates to the 80-20 rule, which comes from the Pareto Index.

### **Now for the code…**

Here is the code for generating a Lorenz Curve:

import numpy as np

expenses = data[data['amount'] < 0]

exp = -expenses['amount'].values

n = len(exp)

lorenz = np.cumsum(np.sort(exp)) / exp.sum()

lorenz = np.append([0],lorenz) # The Lorenz Curve begins at 0

plt.axes().axis('equal')

xaxis = np.linspace(0-1/n,1+1/n,n+1) # There is 1 segment (of size n) for each individual, plus 1 segment at y=0. The first segment starts at 0-1/n and the last one finishes at 1+1/n

plt.plot(xaxis,lorenz,drawstyle='steps-post')

plt.show()

First we select the working sub-sample, which we call expenses. As mentioned above, the individuals must be sorted in increasing order according to the variable’s value; we do it here using np.sort(exp), because exp contains the observations of the “amount” variable. Next, we calculate the cumulative sum using np.cumsum() . To normalize and bring the top of the curve to 1, we divide everything by exp.sum() .

The lorenz variable contains the data point y-coordinates, but now we need their X-coordinates: these run from 0 to 1 (as mentioned previously) in regular intervals. This is what’s generated by np.linspace(0,1,len(lorenz)).

Calculating the Gini Index is a little too complex to go into here, so I will leave it to the bravest among you to look into it further: ^^

AUC = (lorenz.sum() -lorenz[-1]/2 -lorenz[0]/2)/n # area under the Lorenz Curve. The first segment (lorenz[0]) is halfly below O, so we divide it by 2. We do the same for the mast segment lorenz[-1]

S = 0.5 - AUC # area between 1st bisector and the Lorenz Curve

gini = 2\*S

gini

### **Take It Further: Growth Rate**

You often hear about “economic growth,” right? A country’s economic growth is represented by the increase in its Gross Domestic Product (GDP) between year N and the previous year

N−1 .

It is given by growth rate=GDPN−GDPN−1GDPN−1

GDPN is the GDP in the year N )

If you want to express this as a percentage, just multiply it by 100.

For example, if a country has a GDP of $2M in 2020, and the GDP rises to $6M in 2021, its GDP has doubled. The growth rate here will be 200%.

This can be applied to any variable x (in place of “GDP”) and to any time period (in place of the year). If the observed value of variable x at time t is notated xt, then the (empirical) growth rate between moment 0 and moment t is:τt/0=Xt−X0X0

Therefore, if:

* τt/0>0 , variable x has increased between moment 0 and moment t.
* τt/0<0 , variable x has decreased between moment 0 and moment t.

At all costs, please avoid making the following classic error: *“If there is an increase of 100%, followed by a decrease of 100%, we’re back to the initial value!”* In fact, the correct formulation is: ***"If there is an increase of 100%, followed by a decrease of 50%, we’re back to the initial value.”***

It is said that there is no symmetry between increase and decrease.

# **3.6.Check your Knowledge About Performing a Univariate Analysis**

## **QUIZ**

* Perform a Univariate Analysis

**Question 1**

For a given variable, the following values are observed:

{1,10,1,10,1,1,5,1,5,5}

Which of these statements is/are correct? *Please note: there may be more than one correct answer.*

**The Mean is 4.**

**The Median is 3**

**Given data:**

{1,10,1,10,1,1,5,1,5,5}

**Step 1: Mean**

1+10+1+10+1+1+5+1+5+5 / 10 = 40 / 10 = 4

✔️ The **Mean is 4** ✅

### **Step 2: Median**

Sort the data:

{1,1,1,1,1,5,5,5,10,10}

Since there are **10 values** (even count), the median is the average of the 5th and 6th values:

1+5 / 2= 3

✔️ The **Median is 3** ✅

### **Step 3: Mode**

* 1 appears **5 times**
* 5 appears **3 times**
* 10 appears **2 times**

The mode is **1** (since it appears most frequently).

❌ **The Mode is 5** → FALSE

**Question 2**

**Of the following two distributions:**

A={6,4,6,4,6,4,6,4}

B={1,4,1,4,1,4,1,4}

**Which has the greatest variance?**

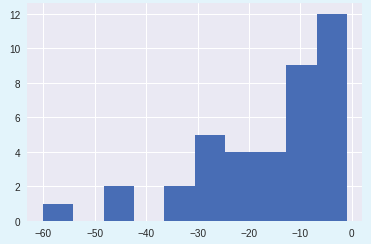
**B**

**Question 3**

**“Skewness” is a measure:**

**of shape**

**Question 4**

****

Take a look at this graph. How could you describe its skewness?

**Negatively skewed; Skewed left**

**Question 5**

**On a given day, in a given region, if it only rains in one town and doesn’t rain in any of the others, what will the Gini Index be for the variable “rainfall” in the sample containing all of the towns of the region?**

**1**

**Question 6**

What statement is FALSE about the Lorenz Curve?

**It is a measure of central tendency**

**Question 7**

Which of the following is a **measure of dispersion?**

**Empirical Variance**

**Standard Deviation**

**Interquartile Range**

**All of the above**

**Question 8**

Which of these 3 measure of central tendency is the most sensitive to outliers.

**The mean**

# **4.1.Understand the Purpose of Bivariate Analysis**

You are now capable of analyzing all of the variables sequentially, one after the other. Bravo, you are an expert at **univariate analysis**!

I’ve analyzed all of the variables! It’s all good! So why are there still chapters remaining in this course? :o

Sorry: it’s not over. In this Part, we are going to analyze the relationships between variables. This is known as **bivariate analysis**. Some chapters will require you to really put in some effort. But if you’ve made it this far, you shouldn’t have anything to worry about. And starting now, analyzing your account statements is going to get really interesting!

### **Why Do We Need Bivariate Analyses?**

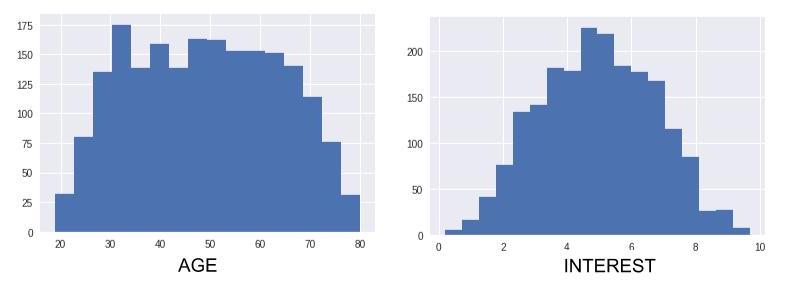
Why should we analyze the relationships between variables?

Here’s a little illustration. Say you work for an e-commerce website. You have access to the site’s customer database and browsing data. The browsing data tell you which customer consulted which page on the site, how much time he or she spent there, etc. To assist in creating a recommendation algorithm (for suggesting new products to customers), you decide to conduct a little preliminary study.

Using the browsing data, you select a sample of customers who often check out the latest folk music albums. You decide to determine how interested they might be in the latest album by a popular singer, modeling this interest using a score of 0 to 10 on a continuous scale. If a given customer has never visited the page for this new album, you assign her an intermediate score of 5. If she has often visited the page, spent a lot of time there, and even ended up buying the album, you assign her a score of 10. If, on the other hand, the customer visited the page, didn’t spend much time there, and didn’t buy the album the last time she ordered something from the website, it’s likely that this customer doesn’t like the new album. So you assign her a score of 0.

You know the age of each customer. So you gather a customer sample defined by two variables: age and level of interest.

You analyze these two variables separately, using histograms:



Histograms: age and interest

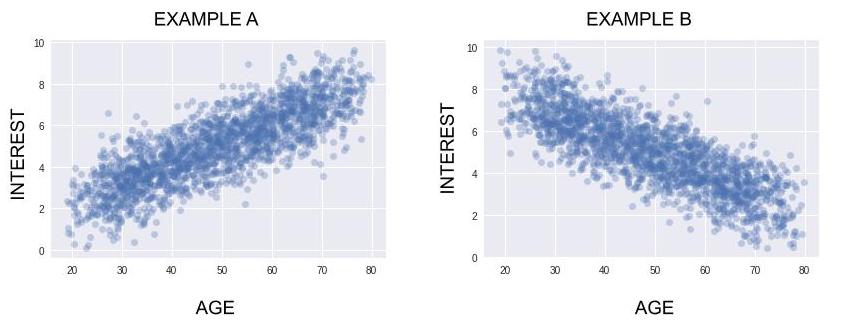
These histograms show that ages are fairly evenly distributed over the sample, which contains approximately the same number of younger people as older people. With respect interest, there are also about as many people who are interested in the new album as there are people who show no interest in it.

Okay. It’s good to know these things but, as we will soon see, we can do much better!

Now let’s plot the individuals of our sample on a two-dimensional graph. Each point on the graph represents a person. The position of each point is determined by two axes: **horizontal** and **vertical**. The values represented on the horizontal axis are **numbers**. If an individual’s number is high, the point will be far to the right on the graph, but if it is close to 0, it will be far to the left. The values represented on the vertical axis are also numbers: if an individual’s number is high, the point will be very high on the graph, but if it is close to 0, it will be very low.

Here, we are representing the *age* variable on the horizontal axis and the *level of interest* variable on the vertical axis. A point located on the upper right will therefore represent a somewhat older person who is very interested in the new album. A point on the lower left will represent a younger person who doesn’t like the album.

Here are two extreme examples:



Notice that neither of these graphs contradicts the findings in the histograms above!

**In Example A**, a lot of older people like this new album, and a lot of younger people don’t like it. So your algorithm should recommend this new album to people who are somewhat older, and it should not recommend it to people who are younger (it’s better to recommend products to people who are predisposed to like them).

**In Example B**, the opposite is true. The album should be recommended to younger people and not to older people.

These graphs are called *“dispersion diagrams”* or *“scatter plots.”*

As I’m sure you’ve realized, in general we get a lot more information from analyzing the relationship between two variables than by analyzing them separately! Without this bivariate analysis, you would not know who should receive the recommendations (or not receive them) for this album!

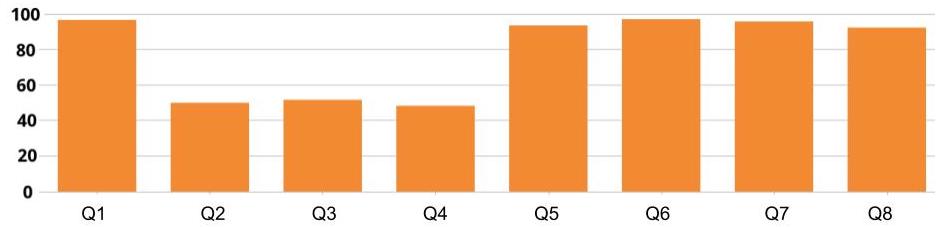
Analyzing the relationship between these two variables is equivalent to asking the following question: *Will knowing that a person is younger (or older) tell me whether he or she is more likely to be interested in this new album?*

### **Take It Further: Another Example**

Another example: Let’s say that a famous online training website publishes courses that require students to take quizzes (sounds familiar? ;)).

To pass a quiz, students must get 70% of the questions right. So, on a quiz of 8 questions, students must answer at least 6 questions correctly in order to pass. *The sample of students who have taken the quiz has 8 variables (each question is a variable).* They are all binary (right answer /wrong answer).

For one of the quizzes in the course, these 8 variables are represented below:



Q1...Q8 means Quiz Question 1....Quiz Question 8.

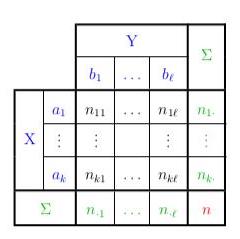
Five questions of the 8 have a success rate of close to 100%. The other three questions have a success rate of close to 50%. This graph represents eight univariate analyses. But we need to analyze the relationships between these variables.

That’s right: of the 50% of students who missed question 2, I don’t know how many got question 3 right, and that’s a problem because:

* If the 50% of students who missed question 2, the 50% who missed question 3, and the 50% who missed question 4 are all the same students, it means that a total of 50% of students failed the test (each with three wrong answers). In this case, the overall pass rate for the test would be 50%, and we would need to simplify the wording of the quiz.
* However, if the 50% of students who missed question 2 are among the 50% who got question 3 right, then these students probably all passed the quiz (because, no matter how they answered question 4, they will almost all have an overall score of 6/8 or 7/8). In this case, the overall pass rate for the quiz would be close to 100%, which is an excellent pass rate!

Here, analyzing the relationships between these variables is equivalent to asking, for example: *Will knowing that a person got question 2 wrong tell me whether he or she is more likely to have given the right answer to question 3?*

# **4.2.Find Correlations in your Data**

****

Contingency Table

By now I’m sure you’ve realized that analyzing the relationships between variables is very important.

The concept of relationship between variables is known more formally as **correlation**. When we say that two variables are correlated, we mean that if we know the value of one, *then* it is possible to have a (more or less accurate) indication of the value of the other.

In the last chapter, when we asked: *“Will knowing the value of an individual’s ‘age’ variable give us a better indication of that individual’s ‘level of interest’ in a given album?”* we were already working with correlations.

### **Causality**

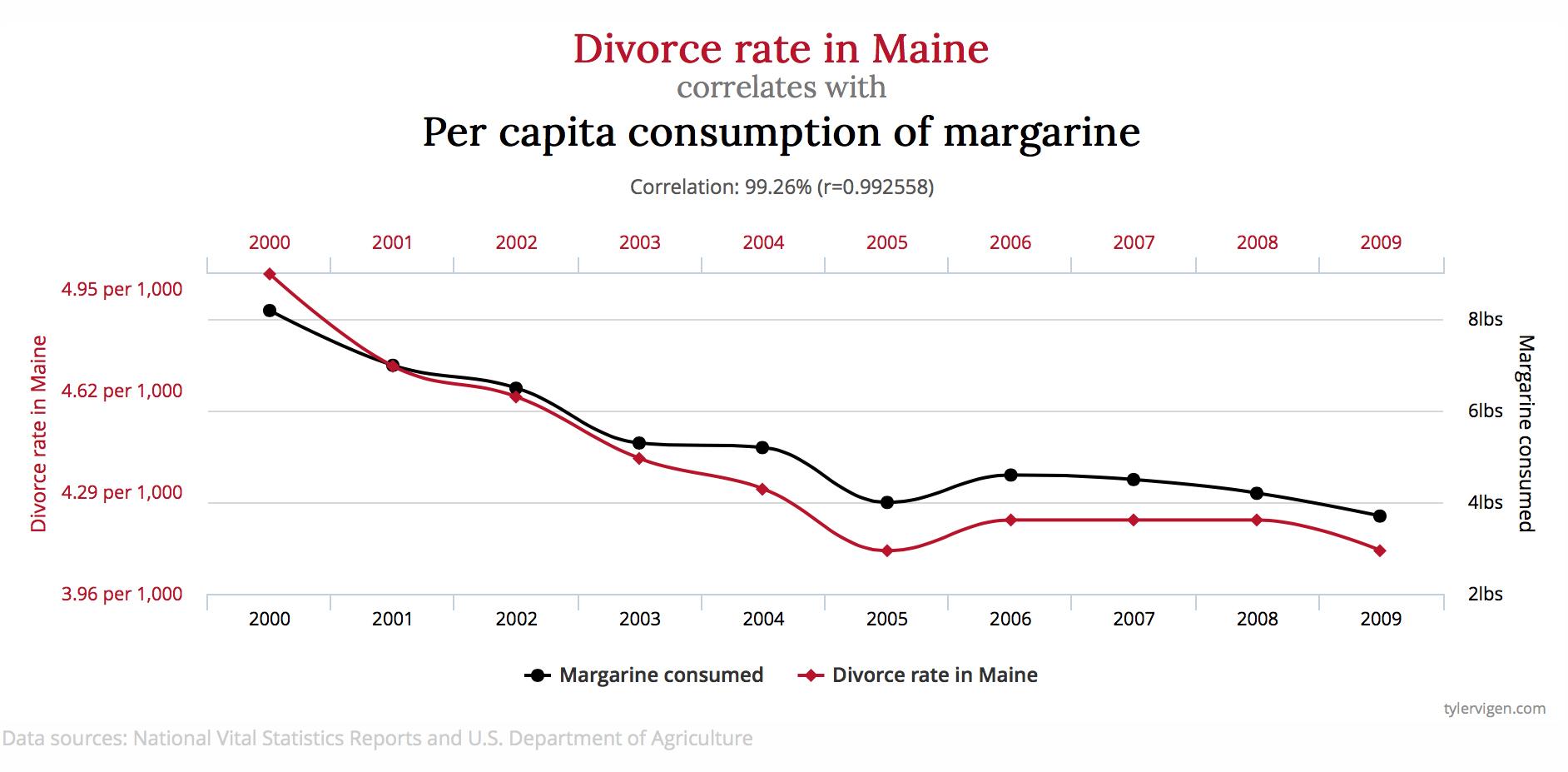
As soon as we beginning talking about correlations, we open ourselves to the possibility of committing a very common error that should **NEVER** be made: that of assuming a cause-and-effect relationship between the two variables in question.

As soon as we beginning talking about correlations, we open ourselves to the possibility of committing a very common error that should NEVER be made: that of assuming a cause-and-effect relationship between the two variables in question.

If there is a correlation between a variable A and a variable B , is A the cause of B , or B the cause of A? It is often impossible to know without conducting an experiment. Most often, the cause of A and B is actually a third (or fourth, or fifth...) factor C that isn’t even always observed.

It is also possible for two variables to be correlated without there being any relationship between them at all.

These are referred to as **spurious correlations**.



To see others like this, [**click here**](http://www.tylervigen.com/spurious-correlations). Some of them are pretty funny!

In fact, before we have the right to assume a cause-and-effect relationship between variables, we need to develop an experiment that meets specific conditions. And if your sample does not derive from this specially designed experiment, these conditions are generally not met.

If you like mathematical paradoxes, here is one that addresses cause and effect relationships: it’s called Simpson’s Paradox.

You’ll see: this paradox will blow your mind!

### **Thinking It Through...**

As I do often in this course, I am now going to appeal to your imagination. It’s always good to use your imagination.

Today, you have a sudden desire to gather statistics concerning the people who live in your town. You want to know which of the following drinks they prefer: coffee, tea, water, or other.

So you investigate, going to cafés, quietly observing customers, and making a note of which drink they order. You collect data on a sample of 100 people. For each person, you record the drink they order and the name of the café in which you observe them. We will call these two variables *café name* and *preferred drink*. Here is the distribution you obtain for the variable preferred drink:

* coffee : 50 people out of 100, or fcoffee = 50 %
* tea : 30 people out of 100, or ftea = 30 %
* other beverages : 20 people out of 100, or fother = 20 %

You continue your investigation, going to a café where you find 10 customers. How many people do you expect to see drinking tea? Intuitively, you expect to find 3 people drinking tea, because you know that, in general, 30% of people order tea. You have therefore made the following calculation: 30% x 10 = 3. Similarly, you expect to see 5 people drinking coffee, and 2 people with other drinks.

But to your great surprise, there are in fact 9 people drinking tea, and only 1 drinking coffee! This is very different from what you expected: 90% of people are drinking tea. Perhaps it’s just chance, you say, so you come back regularly to see if this 90% rate holds steady over time. And, it does: the percentage remains more or less the same after numerous observations!

However, it isn’t long before you understand why: the name of the café is “Chez Louise Tearoom.” This café is a bit out of the ordinary: it’s a tearoom! The customers who come here are primarily tea-lovers.

In a situation like this, we say that *liking tea* and *frequenting the Chez Louise Tearoom* are **not independent** of one another. If two events are **not independent** of one another, we expect to find a correlation between them. Do you recall the question we ask when we’re looking for correlations? *Does knowing the value of one variable give us a more accurate indication of the value of another variable?*

Will knowing that a person frequents the Chez Louise Tearoom give us a more accurate indication of his or her preferred drink?

The answer is **yes**! When we don’t know the value of the *café name* variable, we assume that the preferred drink variable will adhere to this distribution: 50% for coffee, 30% for tea, and 20% for other drinks. **HOWEVER**, when we know the value of the *café name* variable (here: Chez Louise Tearoom), we have a better indication of what the value of the preferred drink variable will be. In this case, we expect to find a lot more than 30% x 10 = 3 people drinking tea.

Let’s label the event *“preferring tea”* I, and the event “being at the Chez Louise Tearoom” J: Here is what we need to remember:

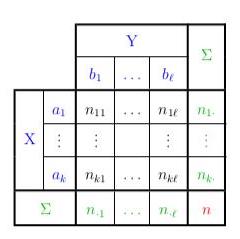
If two events I and J are independent of one another, we expect the number of individuals that satisfy both I and J (let’s call that number nij) to be equal to fi∗nj (this is the calculation you made at the outset: 30% x 10 = 3). On the other hand, the more nij differs from fi∗nj , the more reason we have to believe that I and J are not independent.

When you saw that nab was in fact equal to 9, you realized that “being at the Chez Louise Tearoom” was not independent of liking tea!

### **Learn About Contingency Tables**

We can summarize all of this in something called a **Contingency Table** (where X = *café name* and Y = *preferred drink*):

|  | Coffee | Tea | Other | Total |
| --- | --- | --- | --- | --- |
| Chez Louise | 1 | 9 | 0 | 10 |
| Well Ground Coffee | 9 | 6 | 5 | 20 |
| Rendezvous Café | 20 | 10 | 10 | 40 |
| Sarah's Kitchen | 20 | 5 | 5 | 30 |
| **TOTAL** | 50 | 30 | 20 | 100 |



General Contingency Table

In the first row, we see the number of occurrences of Chez Louise: 1, 9 and 0. In the final row, it is also easy to determine the frequencies for the overall population: 50% (coffee), 30% (tea), 20% (other).

Let’s wrap up this chapter with a little terminology:

* Each value of a contingency table (excluding the TOTAL columns) is called a j**oint occurrence**
* All of the joint occurrences taken together are referred to as the **empirical joint distribution** (*café name*, *preferred drink*).
* The final row (TOTAL) is referred to as the **empirical marginal distribution** of *preferred drink*, and the final column (TOTAL) is referred to as the **empirical marginal distribution** of *café name*.
* All of the joint occurrences in the first row (Chez Louise) are referred to as the **empirical conditional distribution** of *preferred drink*, given the value “Chez Louise” for the variable *café name*.

# **4.3.Analyze Correlations between Two Quantitative Variables**

So far, we have looked at two ways of presenting data in a bivariate analysis: scatter plots and contingency tables.

Scatter plots are useful when both variables are quantitative; contingency tables are useful when both variables are qualitative.

What about when one variable is quantitative and the other is qualitative?

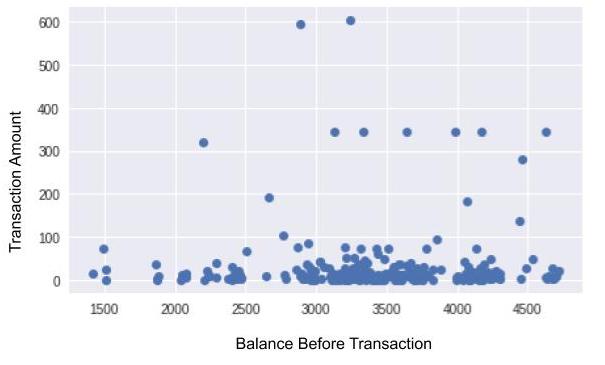
Good for you! You’ve hit on the third possibility! There are four chapters remaining in this course, in which we will analyze each of these cases. This chapter and the one that follows will be devoted to the analysis of two quantitative variables. They will be followed by a chapter devoted to the analysis of one quantitative variable and one qualitative one. The final chapter will focus on analyzing two qualitative variables. Ready? Here we go!

### **Two Quantitative Variables: Graphs**

Let’s ask ourselves the following question:

Do I spend less when I have less money in my account?

As you may have guessed, the two variables to be analyzed here are: *amount* and *balance\_bef\_transaction*. Looking for a correlation between these two variables is equivalent to asking: *"If I know that my account balance is low, can I expect my transaction amount to also be low?"* (or vice-versa)



At first glance, this dispersion diagram doesn’t appear to show that transaction amounts are particularly small when the balance is low. There doesn’t seem to be any correlation (although you may find one in your own statements!). However, there are many points and they are fairly spread out. It is therefore difficult to tell. To remedy this, there is a type of representation that may be better. You’ll find it in the *Take It Further* section at the end of this chapter!

### **Two Quantitative Variables: Numerical Indicators**

Graphs are nice, but I sense that you are missing calculations! We need a numerical indicator that can tell us whether our variables are correlated.

Here, we want to know if, when we have a low balance (X), we also have a small transaction amount (Y). But small in relation to what? Here, when we say “small,” we mean in relation to other values, so what we’re saying is: “below the mean.” Let’s take a random banking transaction (= an individual), and use x to refer to the balance before a transaction, and y to refer to the amount of the transaction. To measure whether x is below the x¯ mean, we can calculate as follows:

x−x¯

If x is below x¯ , this amount will be negative; if it is above x¯, it will be positive. Similarly, to compare y to the y¯ mean , we can calculate y−y¯ . Now, let’s multiply them!

\(a=(x - \overline{x})(y-\overline{y})\)

If \(x\) is below the mean, and \(y\) is below the mean, both will be negative. When we multiply two negative numbers, we obtain a positive number. This also works in the other direction: if \(x\) is above the mean and \(y\) is above the mean, then \(a\)will be positive number.

So with this multiplication, we obtain the amount a for a single banking transaction (a single individual). But if the amounts are truly small when the balance is low (and vice-versa), the as of all of the transactions will be positive! And if we find the mean of all of these as, we will still get a positive number. The mean of all of these *a*s is written as follows:

sX,Y=1n∑i=1n(xi−x¯)(yi−y¯)

On the other hand, if there is no correlation between the balance and the amount, when x is low, y will not necessarily be low. y will sometimes be small, sometimes large, so (y−y¯) will sometimes be positive and sometimes negative. Therefore, a will also sometimes be positive and sometimes negative. Finally, the mean of all of the a s will be close to 0.

To conclude: if x is low when y is small (and vice-versa), then

sX,Y will be positive. If, on the other hand, X and Y are not correlated, sX,Y will instead be close to 0. The more motivated among us will also deduce that if x is *high* when y is *small* (and vice-versa), then sX,Y will be negative. In this case, there is a correlation, of course, but it is referred to as a *negative correlation*.

### **The Empirical Covariance and the Correlation Coefficient**

Guess what! The indicator we just developed is commonly used in statistics; it’s called the **Empirical Covariance** of X and Y. Does this term remind you of the Empirical Variance? It should: they are similar. That’s right, if you calculate the empirical covariance of X and Y, you will find yourself using the formula for the empirical variance of X, which is written

sX2=1n∑i=1n(x−x¯)2 .

Magic!

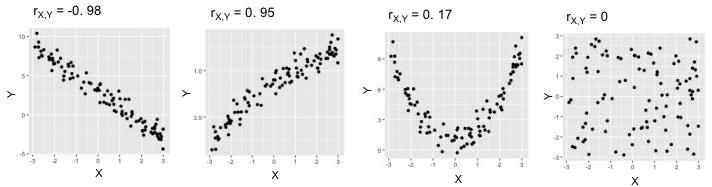
To bring the empirical covariance to a value of between -1 and 1, we can divide it by the product of the standard deviations. This normalization enables us to make comparisons. This gives us:

rX,Y=sX,YsXsY

This **r** coefficient is referred to as the **correlation coefficient**, the **linear correlation coefficient**, or sometimes **Pearson’s correlation coefficient**.

Why *"Linear"*?

Because, unfortunately, it only detects relationships when they are linear, that is, when the points line up more or less in a straight line. In the plots below, the two on the left show points in fairly straight lines: their rs are therefore close to 1 or -1. In the fourth plot, however, there is no real correlation (knowing the x value of a point gives us no indication as to its y value) and therefore r is close to 0. In the third plot, there is a strong correlation, but its shape is not linear; therefore r is also unfortunately close to 0.



### **Now for the code…**

Here is the code for generating the scatter plot seen at the beginning of this chapter:

import matplotlib.pyplot as plt

expenses = data[data.amount < 0]

plt.plot(expenses["balance\_bef\_trn"],-expenses["amount"],'o',alpha=0.5)

plt.xlabel("balance before transaction")

plt.ylabel("expense amount")

plt.show()

Now, to calculate Pearson’s coefficient and the covariance, you need only two lines!

import scipy.stats as st

import numpy as np

st.pearsonr(expenses["balance\_bef\_trn"],-expenses["amount"])[0]

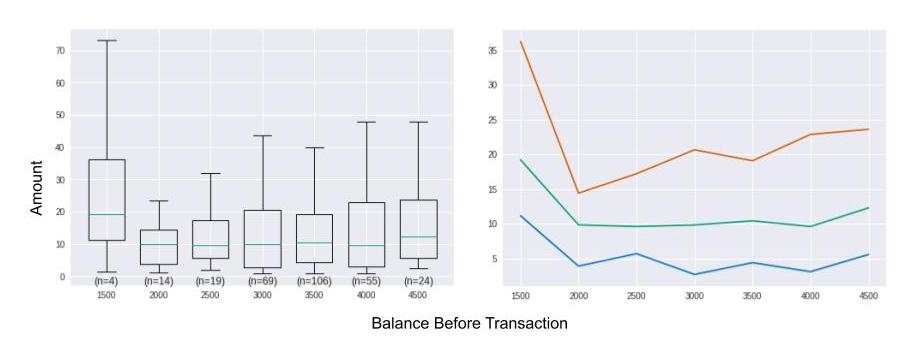
np.cov(expenses["balance\_bef\_trn"],-expenses["amount"],ddof=0)[1,0]

The linear correlation coefficient is calculated by calling the st.pearsonr method. We then give it the two variables to be analyzed. Note that, in this chapter, we prefer to make our expense amounts positive, which is why there is a minus sign - before expenses["amount"] . A pair of values is returned. The first member of this pair is the correlation coefficient, which is why there is a[0] at the end of line 4.

The np.cov method returns the *covariance matrix*, which you don’t need to know about at this level. This matrix is in fact a table, and in this table, it is the value in the second row of the first column, which is why the code contains [1,0] .

### **Take It Further: Alternative to Dispersion Diagrams**

To obtain a clearer representation than a *scatter plot*, we can aggregate the X variable into interval bins along the horizontal axis. This is equivalent to “cutting” the previous graph into vertical slices. For each slice, we display a box plot calculated from all of the points in the slice. Here is the new graph we obtain:

In the graph on the left, we can’t really tell whether the lower balance is, the smaller the amount will be, even though the box plots for slices [2000;2500[ and [2500;3000[ seem slightly denser at the top. The box plot furthest to the left is very dispersed; this may seem surprising, but in fact it is not very representative, because it represents only 4 individuals out of a population of almost 300. Note, therefore, that it is important to display the occurrences for each bin (n=4, n=14, n=19, etc.).

On the right we see an equivalent graph representing the 1st, 2nd, and 3rd quartiles for each slice (bin). This graph can be superimposed on the box plots, because the three quartiles are also visible in the box plots.

bin\_size = 500 # size of bins for discretization

groups = [] # will receive the aggregated data to be displayed

# slices are calculated from 0 to the maximum balance in increments of bin\_size]

slices = np.arange(0, max(expenses["balance\_bef\_trn"]), bin\_size)

slices += bin\_size/2 # slices are separated by half a bin size

indices = np.digitize(expenses["balance\_bef\_trn"], slices) # associates each balance with its bin number

for ind, tr in enumerate(slices): # for each slice, ind receives the slice number and tr the slice in question

amounts = -expenses.loc[indices==ind,"amount"] # selects individuals for the ind slice

if len(amounts) > 0:

g = {

'values': amounts,

'bin\_center': tr-(bin\_size/2),

'size': len(amounts),

'quartiles': [np.percentile(amounts,p) for p in [25,50,75]]

}

groups.append(g)

# display box plots

plt.boxplot([g["values"] for g in groups],

positions= [g["bin\_center"] for g in groups], # X-axis of box plots

showfliers= False, # outliers are not included

widths= bin\_size\*0.7, # graph width of box plots

)

# displays occurrences for each bin

for g in groups:

plt.text(g["bin\_center"],0,"(n={})".format(g["size"]),horizontalalignment='center',verticalalignment='top')

plt.show()

# display quartiles

for n\_quartile in range(3):

plt.plot([g["bin\_center"] for g in groups],

[g["quartiles"][n\_quartile] for g in groups])

plt.show()

### **Take It Further: Properties of the Empirical Covariance**

Very briefly, here are two properties of the Empirical Covariance:

* sX,Y=sY,X . This is the property of **symmetry**.
* If we create a new variable Z from two variables U and V whose empirical covariance is known, and Z=aU+bV , then sX,Z=asX,U+bsX,V . This is the property of **bilinearity**.

# **4.4.Analyze Two Quantitative Variables using Linear Regression**

In the last chapter, we studied the correlation between two quantitative variables. Let’s continue now with two other variables, also quantitative: *wait* and *amount*.

The *wait* variable is populated only for banking transactions in the GROCERIES category. In a previous chapter, you perhaps added values to the GROCERIES category for some of your banking transactions. If this is not the case, I invite you to go back now and download the *enriched\_operations.csv* sample.

A transaction’s *wait* variable gives the number of days that have elapsed between the current and previous transaction in the GROCERIES category. If you buy groceries every seven days on average, the mean of *wait* will be 7.

What do we expect to find?

In theory, the longer you wait to buy groceries, the more you will need to buy. So we expect that the higher the *wait* value is, the larger the *amount* value will be.

### **The Preliminary Step**

First, we need to calculate the *wait* variable! Here is the code (you don’t necessarily need to understand it):

import datetime as dt

# Sub-sample is selected

groceries = data[data.categ == "GROCERIES"]

# Transactions are sorted by date

groceries = groceries.sort\_values("transaction\_date")

# Expenses are converted to positive amounts

groceries["amount"] = -groceries["amount"]

# Wait variable is calculated

r = []

last\_date = dt.datetime.now()

for i,row in groceries.iterrows():

days = (row["transaction\_date"]-last\_date).days

if days == 0:

r.append(r[-1])

else:

r.append(days)

last\_date = row["transaction\_date"]

groceries["wait"] = r

groceries = groceries.iloc[1:,]

# transactions made on the same date are grouped together

# (groceries bought the same day but in 2 different stores)

a = groceries.groupby("transaction\_date")["amount"].sum()

b = groceries.groupby("transaction\_date")["wait"].first()

groceries = pd.DataFrame([a for a in zip(a,b)])

groceries.columns = ["amount","wait"]

Here we create a sub-sample that contains only transactions in the *groceries* category, and call it... groceries!

### **Let’s Model It!**

But we are going to do better than that: we are going to calculate the average cost for the products you consume in one day, and the speed at which you stockpile products in your pantry! To do this, we will use a model. You’ll see: it’s very powerful.

For the model we’re creating, we are going to make a number of assumptions. First, we will assume that each time you buy groceries, you buy three types of products:

1. Products you will consume before you buy groceries again (food, hygiene products, etc.).
2. Products you will not consume during the period being analyzed (this being the time between the first and last receipts you recorded in the sample): this is your long-term stockpile (jars of jam, frozen foods, etc.).
3. Products that are not consumable (ex: forks, floor cloths, etc.), which you buy only rarely.

Next, we will assume that you consume products every day, and that the cost of the products you consume in one day is more or less consistent.

Let’s refer to the average cost of the products you consume in one day (type 1) as *a*, and to the average cost of the products you don’t consume (type 2 and 3 together, which we assume you buy every time you go shopping) as b . Finally, let’s refer to the number of days you waited since the last time you bought groceries as $*\(x\)*$, and to the amount of the receipt as $*\(y\)*$.

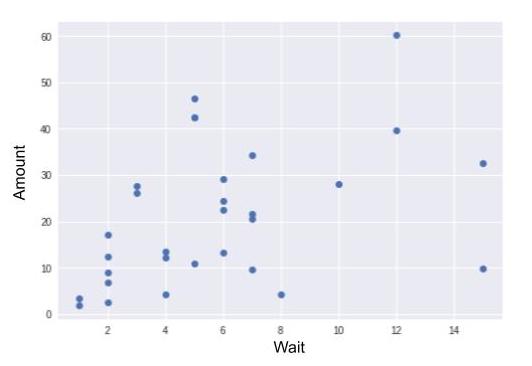
What will the amount of your next grocery receipt be?

It will be equal to the number of days you waited, **multiplied by** the average cost of what you consume in one day. But we also need to factor in the average cost of type 2 and 3 products. This gives the following formula:

y=a.x+b

That’s a little simplistic! My next grocery store receipt won’t be *exactly* this amount. I don’t consume *exactly* the same amount every day, and I don’t buy the same supplies *every time* I go shopping! And I’m sure you realize that I’m going to splurge from time to time!

Yes, our model is simplistic! The equation is not exact. You will also have noticed, no doubt, that it’s a linear equation (remember our affine functions). The fact that it’s a *linear equation* means that if I take all of the possible xs between, for example, 0 and 5, and calculate all of their associated ys before placing them on a graph with the xs on the horizontal axis and lesys on the vertical axis, all of the points will line up perfectly! So let’s try to display a scatter plot where X = *wait* and Y = *amount*, and see if all of the points line up:



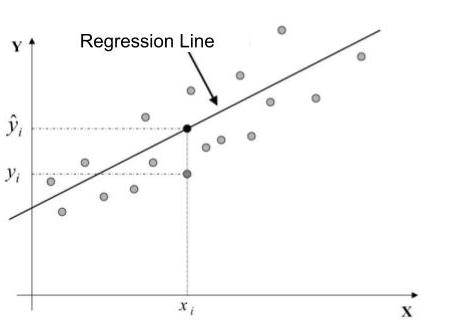
Far from it! This means that the equation y=ax+b is not perfectly accurate: it’s simplistic. With this equation, I admit that there will be an error between my predicted value and the actual value of the next receipt. But I can incorporate this error into the equation, calling it ϵ (*epsilon*):

y=ax+b+ϵ

This is one of the most commonly used models in statistics. It’s known as **linear regression**.

To calculate a and b, I could simply take values at random. But if I did, the error *ϵ* would often be quite large. What I want is to keep my error as small as possible. I am looking to *minimize the error*.

This is how we could represent things graphically. By varying a and b , I shift the line on the graph. Minimizing the error means placing the line in general alignment with the points. The following illustration is for the purposes of instruction; almost all of the points fall into a straight line:



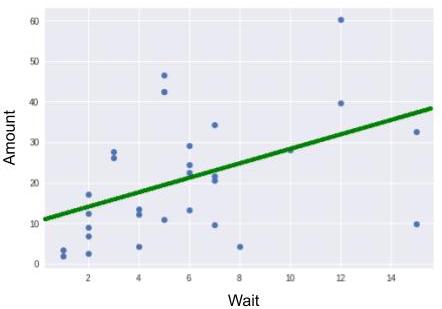
We see that for a point i , we want the difference between yi (which is the true value) and y^i (which is the value predicted by my inexact equation of y^i=a.xi+b ) to be minimal.

There are a number of ways of minimizing error. The most common way is to minimize the Sum of Squared Error

iyi−y^i. It’s called the **Method of Ordinary Least Squares** (OLS).

The computer can estimate a and b for us. To find out more, go to the *Take It Further* section. The following estimates are obtained:

* a^=1.74
* b^=10.94



Regression Line Resulting from Equation y = 1.74x + 10.94

At the beginning of this chapter, we made some assumptions. Simply put, we assumed that there was a linear connection between *wait* and *amount*; that is, a y=ax+b type connection. But is this assumption realistic? After applying a model, you must always analyze its quality. I therefore urge you to read the section *Take It Further: Analyzing Model Quality.*

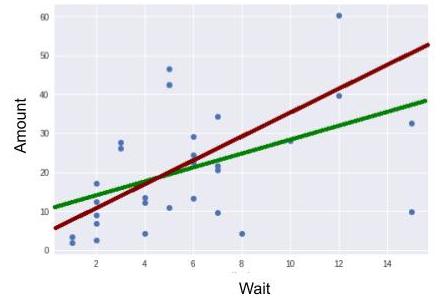
### **Let’s Critique the Result!**

The results indicate that the cost of what I consume daily is only $1.74. That doesn’t seem like much! In addition, $10.94 for long-term supplies every time I buy groceries? That’s huge!

So, it’s true ... looking at it more closely, I see that two points have “broken away from the pack.” We call these *outliers*. Being familiar with my own consumer habits, I know that I never wait more than 15 days to buy groceries. These two outlier points, for which the *wait* value *= 15 days*, correspond, in fact, to times when I had just returned from vacation (during which I did not buy groceries). Since I don’t want these outliers to interfere with my calculation, I discard them.

Once these have been discarded, I obtain the following new estimates:

* a^=3.03
* b^=5.41



This result is quite different from the previous one. Discarding just two individuals significantly altered the outcome. We can therefore conclude that the statistical model we just applied (linear regression estimates using the method of least squares) is **very sensitive** to outliers.

This is also the case, in fact, with last chapter’s linear correlation coefficient *rX*,*Y* : it **isn’t very robust**. In fact, this is no coincidence, because the linear correlation coefficient and linear regression are closely related! To see why, read on!

### **Take It Further: Analyzing Model Quality**

Imagine that I accidentally deleted the amount of a banking transaction in the GROCERIES category.

I could fill in this missing value with the mean of my transaction amounts. This is the most basic solution there is, and as you can imagine, it’s not very good! It’s not very good because *amount* values vary around the mean, sometimes by a lot.

I can do better: I can look at the value of the *wait* variable for this transaction. Using the linear regression model I developed, I can estimate the value (using the equation y=ax+b ). As I’m sure you’ve guessed, this estimate will be better than the one before. That’s right: when we were trying to minimize the modeling error, we were in fact trying to minimize variations in *amount* values around the regression line.

Variations *around the mean* are therefore greater than variations *around the regression line*.

If we had found a perfect model, there would be no error, and no more variations between predicted and actual values. In this case, we would say that the model **explained** all of the variations. Variations around the mean are measured in terms of **variance**. A perfect model would explain 100% of the variations.

This percentage is calculated using the *analysis of variance* formula (ANOVA).

TSS=ESS+RSS

∑j=1n(yj−y¯)2=∑j=1n(yj^−y¯)2+∑j=1n(yj−yj^)2

TSS (Total Sum of Squares) translates the total variation of Y. ESS (Explained Sum of Squares) translates the variation explained by the model. And RSS (Residual Sum of Squares) translates the variation that is not explained by the model.

For linear regression, the explained percentage of variation is given by the **coefficient of determination**, which is written:

R2=ESSTSS

Surprise! From the calculation, we see that R2 is in fact the *square* of the l**inear correlation coefficient** we saw in the last chapter! We have

R2=rX,Y2.

### **Take It Further: Estimating** a **and** b

Here are the formulas used to estimate

a and b:

a^=sX,YsX2 et b^=y¯−a^x¯

Why are there circumflexes over the a and the b ?

This is an estimating convention. It is thought that you can’t directly access your consumer behavior, characterized by a and b, but that you can *estimate* your behavior using your receipts. Estimates of a and b are therefore written a^ and b^ ̂ .

If we added a new receipt to the sample, a^ and b^ ̂ would vary somewhat, even if your consumer behavior didn’t change (that is, even if a and b “didn’t move”).

How do we estimate a and b in the code?

See below. The code here is a little complicated, but remember that line 6 creates the variables a and b that contain the estimates.

import statsmodels.api as sm

Y = groceries['amount']

X = groceries[['wait']]

X = X.copy() # X will be modified, so a copy is created

X['intercept'] = 1.

result = sm.OLS(Y, X).fit() # OLS = Ordinary Least Squares

a,b = result.params['wait'],result.params['intercept']

To display the line, proceed as follows:

plt.plot(groceries.wait,groceries.amount, "o")

plt.plot(np.arange(15),[a\*x+b for x in np.arange(15)])

plt.xlabel("wait")

plt.ylabel("amount")

plt.show()

Line 1 displays the scatter plot.

On line 2, np.arange creates a list of whole numbers from 0 to 14: [0,1,2,3,4,5,6,7,8,9,10,11,12,13,14] .

We place this list on the X-axis. We calculate the ordinates for each of these 15 values using the formula y=ax+b: [a\*x+b for x in np.arange(15)] . In so doing, we create a series of points, all of which line up around the straight line of the equation y=ax+b. Line 2 displays all of these points, arranging them in a nice little line!

# **4.5.Analyze One Quantitative and One Qualitative Variable using ANOVA**

In the last two chapters, we studied correlations between two quantitative variables. Now we’ll move on to two more variables, one of which is qualitative, the other quantitative.

### **Questions We Can Ask**

Depending on which variable pair we choose, the method of analysis will be the same, but there are many different questions we can ask:

* Are the purchases I make on weekends larger than the purchases I make during the week? (variables *amount* and *weekend*)
* Are my purchase amounts larger at the beginning of the month than at the end of the month? (*amount* and *quart\_month*)
* Do my transaction amounts vary from one expense category to the next? (*amount* and *categ*)
* Are my charge payments always small and my deposits always large? (*type* and *amount*)
* Is the balance in my account lower at the end of the month than at the beginning of the month? (*balance\_bef\_transaction* and *quart\_month*)

### **Graphs!**

Here is the code for representing a quantitative variable and a qualitative variable. First, create your working sub-sample, adapting the code, in particular the X and Y variables, to the question you selected from the list above.

X = "categ" # qualitative

Y = "amount" # quantitative

# Only expenses are retained

sub\_sample = data[data["amount"] < 0].copy()

# Expenses are converted to positive amounts

sub\_sample["amount"] = -sub\_sample["amount"]

# Rents are not included because too large:

sub\_sample = sub\_sample[sub\_sample["categ"] != "RENT"]

After which, these six lines of code will display your graph!

categories = sub\_sample[X].unique()

groups = []

for m in categories:

groups.append(sub\_sample[sub\_sample[X]==m][Y])

# Graph properties (not very important)

medianprops = {'color':"black"}

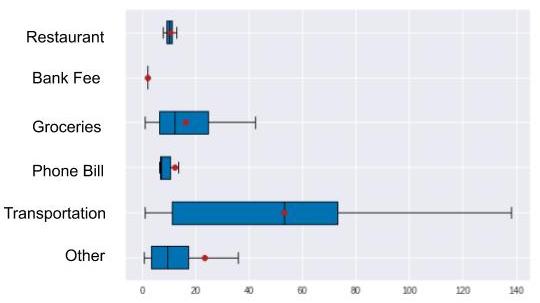
meanprops = {'marker':'o', 'markeredgecolor':'black',

'markerfacecolor':'firebrick'}

plt.boxplot(groups, labels=categories, showfliers=False, medianprops=medianprops,

vert=False, patch\_artist=True, showmeans=True, meanprops=meanprops)

plt.show()



The red dots in the middle of each box plot represent the mean of the values.

Here we see that the amounts vary greatly from one category to the next. For example, transportation expense amounts are higher and more broadly dispersed than phone bill expense amounts. But now let’s verify this with figures, using a model.

### **Model the Data**

Let’s revisit the approach we took in the last chapter. To find out whether there was a (linear) correlation between two variables, we assumed that this correlation existed, then applied a model to this assumption. Next we estimated the a and b parameters. Finally, we verified the initial assumption by evaluating the quality of our model. If our model is high quality, there will be a strong correlation between X and Y. As a bonus, we took advantage of the linear regression formula Y=aX+b+ϵ to interpret aas the cost of what we consume in one day, and b as the cost of our stockpile and non consumable products.We will take the same approach here.

We could use the same linear regression formula as above, but it requires multiplying X by a and, this time, X is qualitative, like our *categ* variable. Multiplying a qualitative variable by a number is not meaningful (ex: “TRANSPORT” x 3 makes no sense!).

So we’re going to take another approach. We will assume that our banking transactions have a common reference amount μ. We will then assume that the transaction amount varies according to the expense category i (rent, transportation, groceries, etc.). If a category contains amounts that are generally lower than μ, the αi adjustment will be negative. If the opposite is true, the adjustment will be positive. We add the requirement that the sum of all of the αis be equal to 0.

For example, a rent amount is generally high: its αrent will therefore be positive.

Just like in the previous chapter, your prediction will always be erroneous, because the amounts are not all the same within the same category!

This is true. And, just as we did with our linear regression model, we will use an error term: ϵ

$*\(Y = \alpha\_{i} + \mu + \epsilon\)*$

As in the previous chapter, the computer can estimate all of the

αi s and μ s, except here, the mathematical calculations that tell us which αi and μ minimize the error ϵ give very intuitive results:

* The reference amount μis estimated using the mean of all of the amounts. We call this estimate μ^*.*
* For a category αi is estimated by calculating the difference between μ^ and the yi¯ mean of the amounts in the category i , that is, αi^=yi¯−μ^.

This model is commonly used in inferential statistics. It is called **Analysis of Variance** (**ANOVA**).

### **Evaluating the Model: Are the Variables Correlated?**

Is ours a good quality model? Are we correctly predicting transaction amounts based only on their category?

As in the previous chapter, we hope that our model is able to explain a high percentage of variations in the data. If it does, it means that there is a strong correlation between our *categ* and *amount* variables .

To evaluate our model, we use exactly the same formula as in the previous chapter: TSS=ESS+RSS. But here, because variable X is qualitative, we can use expressions for TSS, ESS and RSS that are equivalent to those in the previous chapter. They are provided in the *Take It Further* section. These expressions also allow for better interpretation of the three acronyms, which can be renamed, respectively, **total variation**, **inter-classvariation**, and **intra-classvariation** (the classes being the categories of our X variable).

In the last chapter we had R2=ESSTSS . Here we have the equivalent: the **correlation ratio**, between 0 and 1, given by:

ηY/X2=Vinterclass/Vtotal

If ηY/X2=0 , the means per category are all equal. There is therefore theoretically no relationship between variables Y and X. However, if ηY/X2=1, the means per category are very different, each of the categories being made up of identical values: there is therefore theoretically a relationship between variables Y and X.

Here is the code for calculating η2 (*eta squared*). I suggest you perform this calculation by hand: ;)

X = "categ" # qualitative

Y = "amount" # quantitative

sub\_sample = data[data["amount"] < 0] # Only expenses are retained

def eta\_squared(x,y):

mean\_y = y.mean()

categories = []

for category in x.unique():

yi\_category = y[x==category]

categories.append({'ni': len(yi\_category),

'mean\_category': yi\_category.mean()})

TSS = sum([(yj-mean\_y)\*\*2 for yj in y])

ESS = sum([c['ni']\*(c['mean\_category']-mean\_y)\*\*2 for c in categories])

return ESS/TSS

eta\_squared(sub\_sample[X],sub\_sample[Y])

We obtain a result of close to 0.4, which leads us to believe that there is definitely a correlation between the expense amount and the category. Which is what we observed in the graph at the beginning of the chapter!

### **Take It Further: TSS, ESS and RSS Expressions**

TSS, ESS and RSS expressions introduce the occurrences ni of each of the categories i, which are k in number. So:

TSS=∑j=1n(yj−y¯)2 devient ∑i=1k∑j=1ni(yij−y¯)2

Here, **TSS** is referred to as **total variation** (*Total Sum of Squares*).

ESS=∑j=1n(yj^−y¯)2 devient ∑i=1kni(yi¯−y¯)2

Here, **ESS** is referred to as **inter- (or between) class (or category) variation** (*Explained Sum of Squares, or Model Sum of Squares*).

RSS=∑j=1n(yj−yj^)2 devient ∑i=1k∑j=1ni(yij−yi^)2=∑i=1knisi2

Here, **RSS** is referred to as **intra- (or within) class (or category) variation** (*Residual Sum of Squares, or Sum of Squares of Errors*), because si2 is the variance within the category i .

# **4.6.Analyze Two Qualitative Variables using the Chi- Square Test**

Okay... all that’s left now is to analyze two qualitative variables. Let me to reassure you: you did half the work already in the chapter on contingency tables. If you understood that principle, you’re well ahead of the game.

### **Questions We Can Ask**

The method of analysis will be the same no matter which of the following questions you ask. The only thing that changes is the variable pair:

* Do I have the same expense categories on the weekend as during the week? (*categ* and *weekend*)
* Do I have more money coming in at the beginning of the month or at the end of the month? (*debcr* and *quart\_month*)
* Are my expense amounts larger at the beginning of the month than at the end of the month? (*expense\_slice* and *quart\_month*)
* Do my transaction amounts vary from one expense category to the next? (*expense\_slice* and *categ*)
* Are my charge payments always small and my deposits always large? (*type* and *expense\_slice*)
* Are there transaction categories that occur always at the same time of the month, such as rent, for example? (*categ* and *quart\_month*)
* Are there transaction categories for which the form of payment is always the same, for example bank transfer? (*type* and *categ*)

Some questions are identical to those in the last chapter. In that chapter, we used the quantitative variable *amount*, but here, we will use the aggregated variable *expense\_slice*, which is of the same magnitude but is qualitative.

### **Representation**

To answer these questions, you can display a contingency table as follows:

X = "quart\_month"

Y = "categ"

cont = data[[X,Y]].pivot\_table(index=X,columns=Y,aggfunc=len,margins=True,margins\_name="Total")

cont

Replace the two qualitative variables on lines 1 and 2 with those you wish to analyze. The contingency table is calculated using the pivot\_table method. Each cell in the contingency table counts a number of individuals. This count is performed using the len function.

### **Use Statistics**

Unfortunately, we won’t be able to use a model like we did in the last two chapters. But don’t worry! We’ll put things right using a statistical measurement.

Let’s go back to the chapter on contingency tables and look again at what we wrote in the sidebar:

If two events I and J are independent of one another, we expect the number of individuals that satisfy both I and J (let’s call that number nij ) to be equal to fi×nj (this is the calculation you made at the outset: 30% x 10 = 3). On the other hand, the more nij differs from fi×nj , the more reason we have to believe that I and J are not independent.

So, analyzing the correlation between two qualitative variables is the same as comparing the nij s to the fi×njs. The nij s are numbers in the contingency table (except for the TOTAL rows and columns). We could therefore create another table in the same form as the contingency table, but containing the fi×njs instead. Here, then, on the left, is the contingency table that we had before and, on the right, the fi×nj table:

|  | Coffee | Tea | Other | Total |
| --- | --- | --- | --- | --- |
| Chez Louise | 1 | 9 | 0 | 10 |
| Well Ground Coffee | 9 | 6 | 5 | 20 |
| Rendezvous Café | 20 | 10 | 10 | 40 |
| Sarah's Kitchen | 20 | 5 | 5 | 30 |
| **TOTAL** | 50 | 30 | 20 | 100 |

|  | Coffee | Tea | Other | Total |
| --- | --- | --- | --- | --- |
| Chez Louise | 5 | 3 | 2 | 10 |
| Well Ground Coffee | 10 | 6 | 4 | 20 |
| Rendezvous Café | 20 | 12 | 8 | 40 |
| Sarah's Kitchen | 15 | 9 | 6 | 30 |
| **TOTAL** | 50 | 30 | 20 | 100 |

Table 2 is what we would expect if the two variables were independent. So we need a statistic that compares the values of these two tables, and that enables us to pinpoint the pairs of cells whose values really differ. These cells will contain values worthy of interest, and will be the source of non-independence between the two variables.

Here we go! If you want to compare two numbers, I suggest you calculate their difference! And to make sure our differences are always positive (to avoid canceling them out by adding them together), let’s square them. This is not the first time we’ve used this little stratagem:

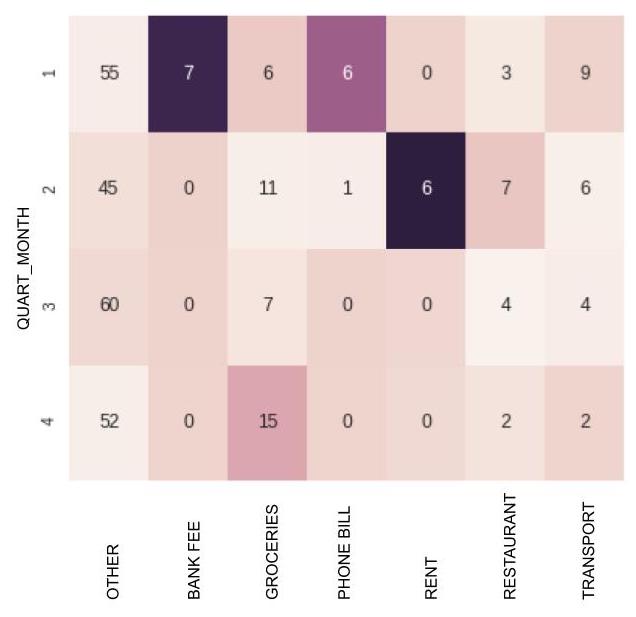
(nij−fi.nj)2

Yes but, if I have nij=2 and fi.nj=4 , the difference when squared will be 4. And if I have nij=1000 and fi.nj=1002, the difference when squared will also be 4. However, an error of 4 when fi.nj equals 4 is a much bigger error than when fi.nj=1002.

That’s true. So we can normalize this difference when squared by dividing it by fi.nj. We then obtain the following formula, knowing that fi=nin :

ξij=(nij−ni.njn)2ni.njn

This measure can be calculated for each cell of the contingency table. It might be interesting to color-code the table according to the results, using dark colors for large results, and light ones for results of close to 0. That way, we can easily tell which cells are the source of non-independence:



Color-coded Contingency Table

Looking at the dark-colored cells, we see that bank fees and phone bills are often paid at the very beginning of the month, that rents are often paid in the 2nd quarter of the month, etc.

Finally, if we add up all of these ξij measures for each cell of the table (from column j=1 to column j=I, and from row i=1 to row i=k), we obtain the statistic ξ is pronounced “zai” or “sigh”):

ξn=∑i=1k∑j=1l(nij−ni.njn)2ni.njn

Normally, we apply a threshold to this measure, above which we say that the two variables are correlated. It’s a little complicated, but here, we simply need to remember that the larger $*\(\xi\_n\)*$is, the less likely it is that any hypothesis of independence will be valid.

The threshold test is called the χ2 Test of Independence ( χ2 is pronounced “kai-squared”), and is sometimes written Chi-2 or Khi-2.

In the last four chapters, we saw the measures r2, η2, and ξn . These are all numbers that provide an indication of the degree of correlation (or its opposite, independence) between two variables.

As we have seen, we can associate each cell of the table with a ξij. We can then normalize each ξij by dividing it by

ξn. We then obtain for each cell a value of between 0 and 1.

This value can be considered a **contribution** to non-independence. Optionally, it can be expressed as a percentage, by multiplying it by 100. The closer this contribution gets to 100%, the more the cell in question will be considered a source of non-independence. The sum of all of the contributions is equal to 100%.

Here is the code for generating the color-coded contingency table:

tx = pd.DataFrame(tx)

ty = pd.DataFrame(ty)

tx.columns = ["foo"]

ty.columns = ["foo"]

n = len(data)

indep = tx.dot(ty.T) / n

c = c.fillna(0) # Null values are replaced by 0

measure = (c-indep)\*\*2/indep

xi\_n = measure.sum().sum()

sns.heatmap(measure/xi\_n,annot=c)

plt.show()

Lines 1 to 6 calculate the indep table, which is the table representing independence. It calls the matrix product (.dot()), which you don’t need to know about at this level.

On line 9, measure contains all of the ξij for each cell of the table. We can then calculate the *contributions* (defined above), dividing each ξij by ξn (placed within the variable xi\_n ). We do this on line 11 with measure/xi\_n. We then obtain a value of between 0 and 1 for each cell.

And that's it! You did! Congrats! You now have all the tools you need to successfully perform an initial data analysis. Don't forget to test your skills in the final activity, and remember to practice, practice, practice the skills that you have learned. Remember, no basketball player ever becomes a great free-throw shooter over night ^^ !

# **4.7.Get Some Practice Performing Bivariate Analyses**

### **It's Your Turn!**

It's your turn! To get some practice, do the following exercise step by step. Once you are finished, you can compare your work with an example I made earlier.

One of the most common data sets used in statistics courses and tutorials is called “Iris.” This data set consists of measurement samples taken from species of the Iris variety. For each flower (individual), four features (variables) were measured:

* petal length
* petal width
* sepal length
* sepal width



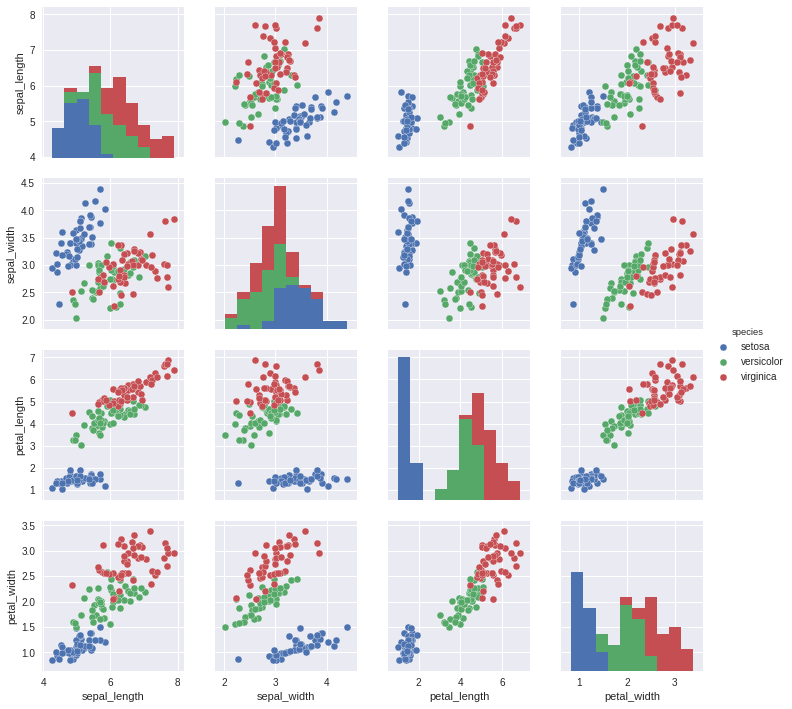
The fifth variable is the species, of which there are three: **Setosa**, **Versicolor** and **Virginica**.

The original data set has been modified; some values have been deleted, resulting in some missing attributes. Other values have been modified.

The goal of this activity is not to find exact values, but to approximate them, using the methods of bivariate analysis you studied in recent chapters.

Download the data [**here**](https://s3-eu-west-1.amazonaws.com/static.oc-static.com/prod/courses/files/parcours-data-analyst/Cours_nettoyez-et-decrivez-votre-jeu-de-donnees/iris_dataset.csv).

Here are scatter plots for the qualitative variables, analyzed in pairs. The points on these diagrams have been color-coded according to Species (a qualitative variable).



Here is the Python code for loading the data set:

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Download

iris = pd.read\_csv("iris\_dataset.csv")

# Rename the columns

iris.columns = ["id","sepal\_length","sepal\_width","petal\_length","petal\_width","species"]

# Delete the identifiers

del iris["id"]

# Delete individuals with at least one missing value

iris\_dna = iris.dropna(axis=0, how='any')

print("iris : {} individuals, iris\_dna : {} individuals".format(len(iris),len(iris\_dna)))

# Show dispersion diagrams

sns.pairplot(iris\_dna,hue="species")

plt.show()

iris\_dna has been divided into three samples, one for each species:

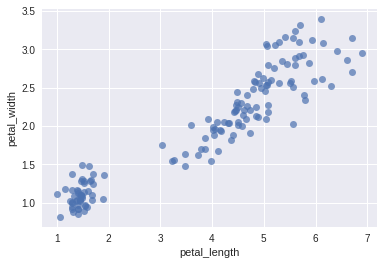
iris\_setosa = iris\_dna[iris\_dna["species"] == "setosa"]

iris\_virginica = iris\_dna[iris\_dna["species"] == "virginica"]

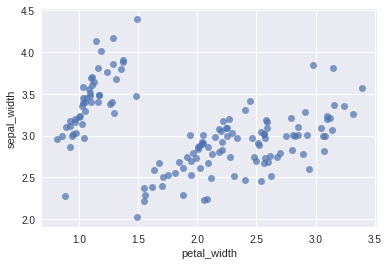
iris\_versicolor = iris\_dna[iris\_dna["species"] == "versicolor"]

Here are two scatter plots:

* one showing petal\_width as a function of petal\_length



* one showing sepal\_width as a function of petal\_width



#### **Questions**

##### **Question 1:**

In the iris\_dna dataframe, calculate the following linear correlation coefficients:

* petal\_width as a function of petal\_length
* sepal\_width as a function of petal\_width

##### **Question 2:**

Explain the results of Question 1 using the two scatter plots provided above in the statement.

Now, bear in mind the linear correlation coefficient of sepal\_width as a function of petal\_width. It has been calculated for all of the irises. If the linear correlation coefficients are calculated for these same variables, but are separated by species, the following results are obtained:

* sepal\_width as a function of petal\_width for iris\_setosa: 0.753
* sepal\_width as a function of petal\_width for iris\_virginica: 0.685
* sepal\_width as a function of petal\_width for iris\_versicolor: 0.825

Please note: You do not need to recalculate.

Compare (in your head) these three results to the result obtained for all of the irises above (Question 1).

Please note: You do not need to provide the answer to this question in the work you submit. The results are easier to interpret when the points are color-coded by iris species:

##### **Question 3:**

You will calculate four linear regressions (using the Method of Ordinary Least Squares). We will use the following notation: Y = aX + b + epsilon .

a and b are the real numbers you need to estimate, epsilon is the term of error (you don’t need to worry about this), X and Y are two variables.

The four linear regressions correspond to each of these four cases:

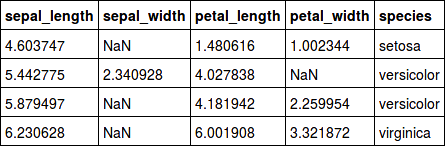
**Case 1: X is the variable petal\_length, and Y is the variable petal\_width, in the iris\_dna dataframe  
Case 2: X is the variable petal\_width, and Y is the variable sepal\_width, in the iris\_setosa dataframe  
Case 3: X is the variable petal\_width, and Y is the variable sepal\_width, in the iris\_virginica dataframe  
Case 4: X is the variable petal\_width, and Y is the variable sepal\_width, in the iris\_versicolor dataframe**

Note: These are the same four examples we have been analyzing from the beginning.

For each of these examples, estimate a and b.

##### **Question 4:**

Here are four rows, each of which contains missing data:



For each individual, the missing attribute is found either in the petal\_width variable, or in the sepal\_width variable. In both these cases, the values can be imputed (filled in) using the linear regressions we analyzed before. These values will approximate the actual values, but will be false.

Assuming that one individual never has more than one missing value, here is the code that uses linear regressions to replace the missing values. Parts of the code have been removed and replaced by “[...]”: it is up to you to fill them in.

coeffs = {

"case 1" : {'a': [...] , 'b':[...]},

"case 2" : {'a': [...] , 'b':[...]},

"case 3" : {'a': [...] , 'b':[...]},

"case 4" : {'a': [...] , 'b':[...]},

}

modified\_lignes = []

for (i,individual) in iris.iterrows(): # for every individual of Iris,...

if pd.isnull(individu["petal\_width"]): #... we test if individual["petal\_width"] is null.

a = coeffs["case 1"]['a']

b = coeffs["case 1"]['b']

X = individual["petal\_length"]

Y = a\*X + b

iris.loc[i,"petal\_width"] = Y # we replace the missing value by Y

modified\_lines.append(i)

print("we filled petal\_width with {} based on petal\_length={}".format(Y,X))

if pd.isnull(individual["sepal\_width"]):

espece = individual["species"]

X = individual["petal\_width"]

[...]

modified\_lines.append(i)

print("We filled sepal\_width with {} based on the species:{} and petal\_width={}".format(Y,species,X))

print(iris.loc[modified\_lines])

#### **Deliverables**

* the Python code you used to answer the questions in .py format. If you wish to submit your work in a notebook, please also submit a version in .py, in case your grader has not installed the program necessary for reading the notebook.
* the answers to the four questions in .txt file format.